NASA Contractor Report 198479

Technology Directions for the 21st Century
Volume II

Giles F. Crimi, Henry Verheggen, John Malinowski,
Robert Malinowski, and Robert Botta
Science Applications International Corporation
McLean, Virginia

May 1996

Prepared for
Lewis Research Center
Under Contract DAEA32–96–D–0001

National Aeronautics and
Space Administration

PREFACE

It has been said that the only thing that is constant is change. This adage is well borne out in both the understanding of space and the pursuit of that understanding. In the last twelve months, the scientific studies of our universe, made possible by spectacularly successful flight programs such as the Hubble Space Telescope, Galileo, and Magellan, almost daily have expanded our understanding of the universe and the planets in our solar system. Here on Earth, the space business has also undergone a similar revolutionary process. In the ever complex requirement to balance the pursuit of meaningful science programs with the shrinking availability of funding, NASA has entered the era of having to do more with less. Our focus has shifted to designing faster, cheaper, and better missions without compromising safety or scientific value. Therefore, our technology strategies must correspondingly change to accommodate this new approach. Technology application and systems acquisition must emphasize standardized, multi-use services and operations concepts, leveraged with those of our industrial partners, international counterparts, and academia, as appropriate. Movement toward privatization of space business will be required to better share rewards, exchange technical knowledge, and increase operational efficiencies. These changes are no longer options, but are essential to sustaining our economy and assuring the continued technological leadership of our Nation.

One common thread to the success of our flight programs has been the reliable communication of each bit of information from the spacecraft to the ultimate user. The responsibility for this critical task primarily has rested within NASA's Office of Space Communications (OSC). OSC provides standardized, multi-use communications, data processing, navigation, and mission operation services to both NASA and non-NASA flight programs. NASA's Office of Advanced Concepts and Technology is chartered to develop and advance technology. OSC is a technology user, which adapts existing technology and applications to meet specific program requirements. While OSC's success has greatly benefited from the explosive growth of communications and computing technologies, this new paradigm of "faster-cheaper-better" will require an even greater ability to predict where technology is going.

These reports focus on supporting one element of this new paradigm: the continuing effort to predict and understand technology trends to better plan development strategies. The objectives of this series of documents are to: (1) validate, revise or expand relevant technology forecasts; (2) develop applications strategies to incorporate new technologies into our programs; and, (3) accomplish Agency goals, while stimulating and encouraging development of commercial technology sources. Volumes I and II, together with a future Volume III, summarize several studies relevant to communication and data processing technologies which were selected for their appropriateness to help us meet our challenges.

Comments from interested parties would be especially appreciated, and may be directed to NASA Headquarters, Dr. Albert R. Miller, at (202) 358-4804, or to NASA Lewis Research Center, Ms. Denise S. Ponchak, at (216) 433-3465.

[illegible]

Charles T. Force
Associate Administrator for
Space Communications

ACKNOWLEDGMENT

This work was performed by Science Applications International Corporation (SAIC), Technology Applications Group, in McLean, Virginia for the National Aeronautics and Space Administration (NASA). This report summarizes a series of investigations of various technologies which are important to long range planning for the operational systems and capabilities for which NASA's Office of Space Communications is responsible.

NASA is pleased to acknowledge the following SAIC personnel who contributed to this effort: Dr. Giles Crimi (Program Manager), Mr. Henry Verheggen (Principal Investigator), Messrs. John Malinowski and Robert Malinowski (Contributing Investigators) and Dr. Robert F. Botta (volume update and Technical Editor).

The technical direction for this work was provided by Ms. Denise S. Ponchak, NASA Lewis Research Center, and Dr. Albert R. Miller, NASA Headquarters, Office of Space Communications.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

FOREWORD

The National Aeronautics and Space Administration (NASA) is a major user of communications and information systems technology. NASA requirements are growing rapidly as missions and instruments become more complex and data processing intensive. NASA information systems typically have long life cycles not only because of the nature of the science missions and the realities of the funding process, but because missions are delayed and expected lifetimes are exceeded. Systems development is complicated by the fast pace of technological change, exemplified by the doubling in performance of microprocessors as quickly as every 18 months, popularly referred to as Moore's Law. Systems may become obsolete early in the life cycle, unless the capability to upgrade is designed in from the beginning. Waiting, for example, 10 years to upgrade a system because of budget constraints may no longer be affordable, since the changes in technology may make it nearly impossible to salvage existing software, a major investment cost. In order to be able to plan for upgrades to major systems, data are needed to project the future performance levels of communications and data processing systems and related components. Ultimately, such data may be used to build a model for technological change that would allow planners to make informed estimates of system cost many years in advance. This would permit a "just-in-time" strategy for technology insertion at the moment in time when it is still at the state of the art.

The Office of Space Communications (OSC) is tasked by NASA to conduct this planning process to meet NASA's science mission and other communications and data processing requirements. A set of technology trend studies was undertaken by Science Applications International Corporation (SAIC) for OSC to identify quantitative data that can be used to predict performance of electronic equipment in the future to assist in the planning process. Only commercially available, off-the-shelf technology was included. For each technology area considered, the current state of the technology is discussed, future applications that could benefit from use of the technology are identified, and likely future developments of the technology are described. The impact of each technology area on NASA operations is presented together with a discussion of the feasibility and risk associated with its development. An approximate timeline is given for the next 15 to 25 years to indicate the anticipated evolution of capabilities within each of the technology areas considered. The steep rate of change projected in some technology areas underscores the need for such data for the purpose of planning information systems. Another beneficial result of the data collection effort is the insight it provides to areas where breakthroughs are likely to occur in key technologies.

The study findings are organized into two volumes. Volume I consists of four chapters covering computers, data storage, and photonics. Two chapters are devoted to computers: one examines the physical level of chips and molecular computing; and the other examines the system level, including classes of processors, system architectures, and applications. One chapter each is devoted to data storage and photonics technologies. This volume contains four chapters: one each on technology trends for database systems, computer software, neural and fuzzy systems, and artificial intelligence. The principal study results are summarized at the beginning of each chapter.

CHAPTER 1. DATABASE TECHNOLOGY TRENDS

SUMMARY

Database applications are sufficiently general and address a large enough market so that it is both feasible and economical to build specialized hardware and architectures to accelerate database processing. Because large databases may have a large number of users, the requirement for processing power and high-speed storage access may be greater than that allowed by current technology. Several technologies are available or on the horizon that could be used to accelerate database processing. These include:

- Parallel arrays of magnetic disks for access time improvement
- Holographic imaging for parallel access of optical disks and photorefractive crystals
- Associative memory for parallel access
- Balanced systems architectures, or architectures in which processing, storage and networking are well-matched for maximum throughput
- Photonic devices and architectures for database acceleration
- Artificial intelligence (AI) technology applied to database search.

1. INTRODUCTION

Databases may be the most common generic application for large computer systems with multiple users. In the near future NASA will become the builder of some of the largest databases in existence, particularly for imagery data from low-earth-orbiting satellites. Typically, storage and networking technology has not kept pace with computer technology for large databases. As NASA's databases move into the petabyte (10^{15} bytes) range, extensive planning will be required for networks and on-line and archival storage systems. These systems will have to be closely matched or balanced – meaning that the design of networks cannot be done independently of storage or processing. NASA will also profit from the use of specialized hardware and software for accelerating the performance of database systems.

2. LONG-RANGE TECHNOLOGY TRENDS

2.1 ANALYSIS OF TRENDS

An emerging trend for database systems results from efforts to build balanced systems, or systems in which bottlenecks between the processor, the various levels of storage, and the network are eliminated. Techniques by which this is being accomplished include the use of parallelism in storage and networks as well as in computing and the use of hierarchical architectures. Another approach is to accelerate search and retrieval speeds by the use of special purpose architectures such as Hopfield associative neural networks and the use of photonics technology. These are discussed in more detail in section 2.3.

An important trend is boosting database search speed by means of parallelism in processors, in storage, and in local area networks (LANs). The use of parallel processors for database analysis is a relatively new development. Retail corporations are among the first commercial purchasers of massively parallel computers. They use them for "database mining", or searching their inventory and transaction databases for hidden patterns in consumer buying [1]. Single-instruction, multiple-data (SIMD) array processors such as the Loral ASPRO have been used by the military for fast tactical database search by means of associative memory techniques (discussed in more detail in section 2.3).

The pursuit of parallelism in computing has resulted in widespread research in the realm of architectures that are scalable to large numbers of computing elements. Integrated circuits have begun to emerge that contain multiple processors per chip. The most favored class of architectures is the class of neural network or connectionist architectures. These are capable of being scaled to high numbers of simple processors. The processors are simple enough to allow many processors with memory to be built on one chip. One such architecture, the Hopfield associative memory, is frequently proposed as applicable to database storage. In this architecture, each stored data word requires a processor. Useful associative networks would require millions of processors per chip, and are therefore somewhat beyond the capabilities of today's technology.

Parallelism for storage is an emerging trend, in particular the use of redundant arrays of inexpensive disks (RAID). A future development is parallel access to storage media by means of parallel optical transducer arrays and holographic imaging. Holographic techniques can also remove another bottleneck, that of the mechanical delays of rotating disk drives.

The introduction of parallelism into LANs is occurring through the use of circuit switching, which creates the topology of a mesh of multiple parallel point-to-point connections. Point-to-point connections allocate more bandwidth to the connection than in a shared medium, and minimize the amount of software protocol processing. Topology-independent LAN standards are emerging, such as the Fibre Channel Standard (FCS). FCS is capable of making use of both packet and circuit switching within the same physical network.

A common theme in these developments is the use of photonics for the transmission of signals at all levels of the system, mainly because of the high potential bandwidth of photon signals (10^{12} Hertz), and the ability of photonic devices to achieve high parallelism with low crosstalk.

2.2 FUTURE APPLICATIONS

Database applications will be a growth industry for the foreseeable future, and demand for ever better performance will parallel that for computer performance in general. However, because of the centralized nature of large databases, technology improvements do not necessarily keep pace with the demand for performance growth. This is because the number of users attempting to access the same information can easily grow faster than the improvement in performance. As the cost of acquiring the data goes down, more applications for the data are opened up. Commercial applications for remote sensing imagery are growing, including for such relatively common purposes as real-estate development planning. If NASA chooses to subsidize the costs of these commercial applications, they will undoubtedly grow rapidly, as has been the experience with the government-sponsored Internet, which has seen growth rates as high as 25 percent per month.

The two main database types are image and text databases. Of these, image databases have the most stressing requirements. NASA image databases will be geographically distributed, and will have users distributed nationwide. For these to be widely used, remote users will want to have the capability to browse through data sets with low retrieval delays. Ideally, it would be desirable to be able to download entire data sets at high speed (hundreds of megabits to gigabits per second). Images might be retrieved using standard forms of indexing based on metadata such as instrument, date, time of day, geographic location, and wavelength. However, an increasingly common form of image search may be that based on content. In other words, algorithms will be developed that will be able to classify and retrieve images based on complex features of interest within the image. An example would be to retrieve only those images having a certain type of terrain as identified by a spectral reflectance signature. This kind of intelligent search would require fast parallel processors dedicated to executing the search algorithms. Statistical pattern recognition and neural network classification methods will be increasingly important for this purpose.

For document and text databases, traditional relational database management techniques have been used, but these techniques were designed for structured data lists, not for free-form text. New techniques are being developed for this kind of database, including both hardware correlators for scanning the entire text at high speed, as well as AI methods for intelligent search, such as document clustering. Some of these methods are discussed in later sections.

2.3 EMERGING TECHNOLOGIES

2.3.1 Parallel Storage

The simplest technique in present-day use for speeding up the input/output (I/O) rate of mass storage is to read and write to multiple disks in parallel, in a configuration sometimes referred to as parallel transfer disks (PTD). RAID products are available which implement the PTD function. RAID can be used both for availability enhancement and transfer rate speed-up. A recent article [2] describes the use of a RAID product built by Storage Concepts, Inc., together with a FCS 266 megabits (10^6 bits) per second (Mbps) LAN offered by IBM/Hewlett Packard for data visualization networking.

Another approach is to do parallel access on a single drive. For example, optical disk storage is well-suited to parallel access. Instead of using a single read/write laser head, a multiple- laser head could be used. The invention of vertical-cavity surface-emitting lasers makes this possible. Hundreds of these microlasers can be manufactured on a single substrate to form a microlaser array. Photonics Research, Inc., for example, has recently begun to market this kind of device. An alternative scheme is to store data holographically on an optical disk, and to use a single unfocused beam to read out thousands of bits in parallel.

Optical holography can be applied to volumetric storage to achieve the ultimate form of parallel access. This technique results in high storage density and fast, parallel, random access. A spatial light modulator steers a laser beam that interrogates holographically-stored bit patterns without the need for a mechanical drive. This beam-switching approach is much faster than the mechanical positioning of magnetic read/write heads. The laser beam also illuminates a whole page of memory simultaneously, allowing parallel readout of the page. Database applications of this technology are discussed by Berra [3]. In mid-1995, a university/industry consortium led by Tamarack Storage Devices received a five-year $11 million grant from the Advanced Research Projects Agency (ARPA) to develop a wave guide holographic storage system [4]. Industry participants will contribute additional funding, bringing the total to $22 million. The fact that Tamarack originally planned to market such a device in late 1993 or early 1994 indicates the difficulty in perfecting this new technology.

2.3.2 Associative Memory

In a conventional random access memory, the content of a storage cell is accessed by its address. An associative or content-addressable memory (CAM) is one in which the content is accessed by means of a given search argument, such as a keyword. This approach is often proposed as a superior means of implementing high-performance, parallel-search database computers. Associative architectures are a form of parallel distributed storage. Research in this field is an attempt to imitate the human faculty of memory, which seems to be associative in nature.

One method of implementing an associative memory is to simultaneously compare the content of all storage locations with a keyword. This is done, for example, in specialized CAM chips developed for network routers, in which each address location has a comparator. This is a relatively expensive approach since the number of comparators drives up the circuit cost relative to the amount of storage [5]. An extensive discussion of very large scale integration (VLSI) CAM and associative processor designs is given by K.E. Grosspietsch [6]. He describes several applications for these recent designs: database support, image pixel processing, emulation of Petri nets, and simulation of neural nets.

Many researchers are investigating architectures for future massively parallel systems in which the number of processing elements is very large and the circuit density is very high. This extrapolation envisions hundreds or thousands of processors per chip. In this kind of architecture the processor must be very simple and have its own memory. Processing and memory functions are not separate in this architecture. A great deal of attention has focused on one particular implementation, the Hopfield neural network [7]. In this network, a data object is stored in a distributed manner over the entire network. For each object, represented by a digital word or vector to be stored, a matrix is created using an outer product operation. The matrices for all the stored words are added together. To retrieve the original stored word, a garbled or partial version of the desired word or an associated keyword is multiplied by the matrix, and the result is the original word. Figure 1-1 illustrates a hardware version of the Hopfield network. In this network the inputs are multiplied by the weights and the results are summed at each node. These sums are fed back through the network, and after several cycles, the desired stored vector appears at the output.

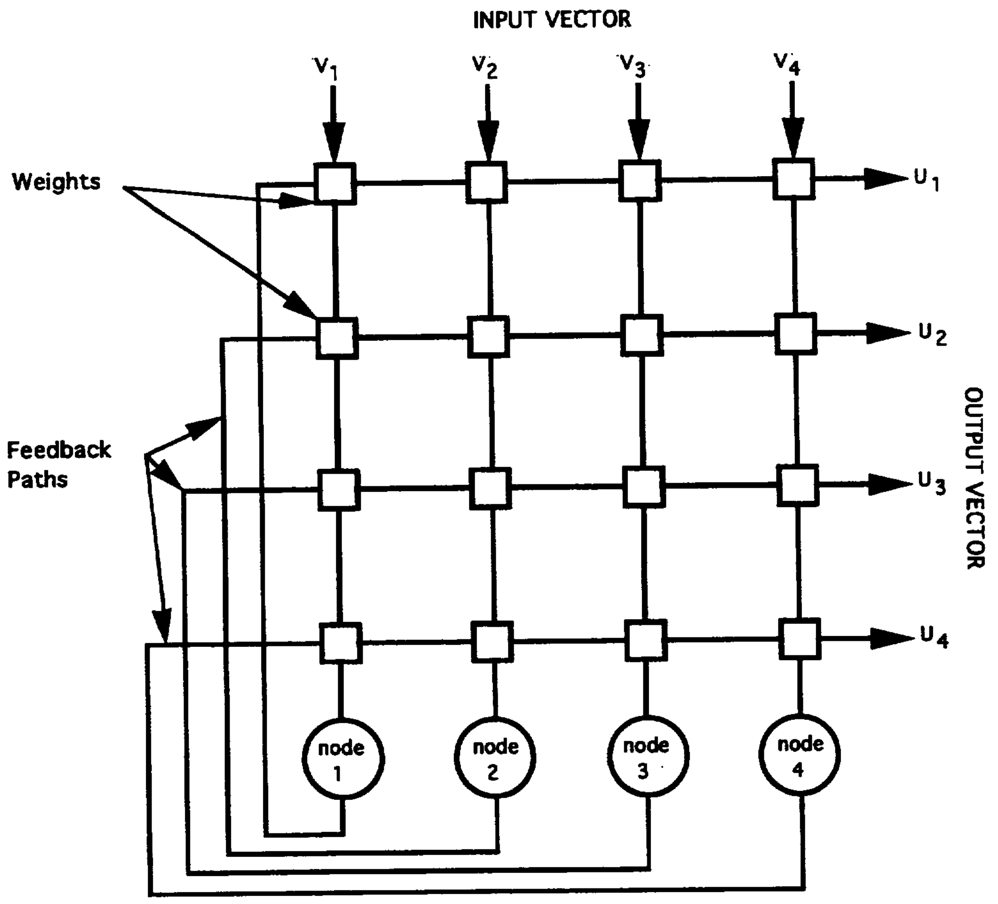

Figure 1-1. The Hopfield Network Architecture

The original Hopfield memory has several practical problems, including limited storage capacity relative to the number of processing elements (neurons), and a storage and retrieval process that is computationally intensive when implemented in digital form. Hopfield estimated the storage capacity of a network of N neurons to be about 0.15 x N. An upper limit of N was later derived theoretically by Y.S. Abu-Mostafa and J.M. St. Jacques [8]. A more elaborate analysis by R.J. Eliece et al. [9] puts an upper bound at N/(4 log N).

Work is being done on improved Hopfield-like memories that require less computation. One such alternative is to use an inner-product method for constructing the storage matrix. Hua-Kuang Liu of Caltech under contract to the Jet Propulsion Laboratory (JPL) reports on an optical computer using this approach [10]. Hardware implementations of the Hopfield architecture are being built at many research establishments. JPL researchers have built VLSI Hopfield associative memory chips that demonstrate fault-tolerance [11]. The fault-tolerance property is a direct consequence of the ability of the network to converge to the correct state when some bits are in error. This feature could allow the memory to be constructed more densely than in conventional architectures. Others are working on optical implementations, some of which are described below.

2.3.3 Optical Database Computers

VLSI systems are limited in their ability to implement parallel architectures because of limitations in the number of possible I/O pins, the off-chip bandwidth, and the on-chip interconnection density. Optical computers are often proposed for high throughput applications because they achieve a potentially larger number of parallel I/O channels by making use of a third dimension for interconnection. Recently, a digital optical computer for database applications called the DOC II was built by OptiComp Corp [12]. This computer implements a RISC instruction set and uses 64-channel acousto-optic modulators for binary spatial light modulation. The basic method of implementing a Boolean operation is illustrated in Figure 1-2. If the signal modulating the laser carries a logic-1 and the signal carried by the acousto-optic modulator carries a logic-1 in the same clock cycle, the light will be diffracted by the acoustic wave and will come to a focus on the detector. It is essentially a free-text search processor, and performs 12.8 billion character compares per second. Free-text search is useful for text databases that are not indexed, which because of the high cost of indexing is becoming an increasingly important alternative. Although this computer apparently at least equals the performance of present-day digital electronic supercomputers, it is not clear that it is actually competitive, since an equivalent electronic symbol-string correlator could presumably be built relatively straightforwardly using high-speed correlator chips. However, work on the next generation of this computer has started. In the next version, an advanced spatial light modulator (SLM) will be used that has the potential to increase the throughput from 1 trillion (10^{12}) bit operations per second to 100 trillion (10^{14}) bit operations per second [13].

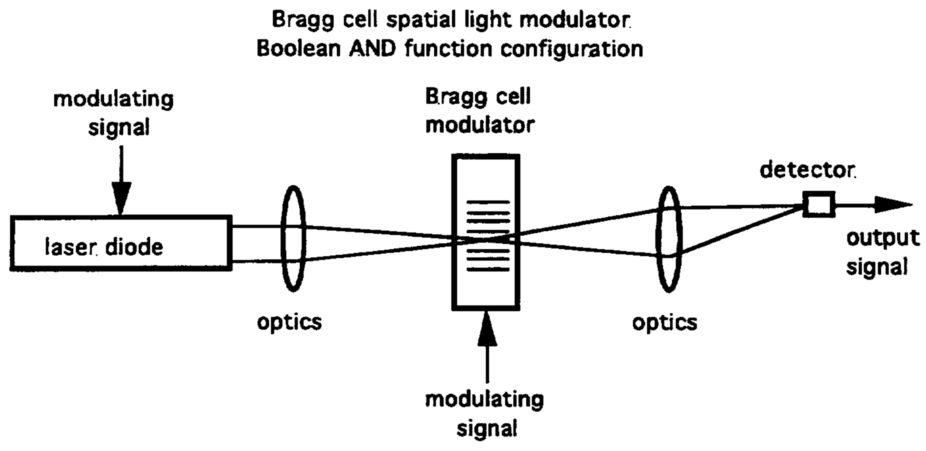

Figure 1-2. DOC II Principle of Operation

Many other architectures for optical database computers have been proposed, based on the high throughput capability of acousto-optic SLMs and the matrix-processing capabilities of 2-

dimensional SLMs and lenses. Y.S. Abu-Mostafa and D. Psaltis have described work on an optical implementation of a Hopfield associative memory for image database applications [14]. This demonstration uses 2-dimensional SLMs as input-output devices and photographic film holograms as the data storage elements. Another such implementation is described by Farhat et al. [15] in which only lenses and light-emitting diode (LED) arrays are used. The matrix-multiplication properties of lenses are used to perform the Hopfield outer-product matrix operations. The matrix-multiplication properties of lenses can be used directly for optical symbol correlation for text-search processors as described by Berra [op. cit.] More recently, Hua-Kuang Liu [op. cit.] used an inner-product associative memory method to construct an optical computer. This computer uses liquid-crystal spatial light modulators.

2.3.4 Pyramid and Hierarchical Architectures

A pyramid architecture is a parallel computer architecture specialized for image processing, and thus is relevant to image database processing. In one such architecture, the "image understanding array" architecture [16] there are three levels of arrays of parallel processors. As illustrated in Figure 1-3, the lowest level consists of a SIMD array of 512 by 512 single-bit pixel processors. This level deals with pixel operations on the raw image data. The second layer consists of an array of 64 by 64 16-bit processors which operate on the results of the first layer. The final layer is a set of 64 32-bit processors that is supposed to execute syntactic-AI processing on abstract features defined by the first two layers. Although this particular system is to be implemented using

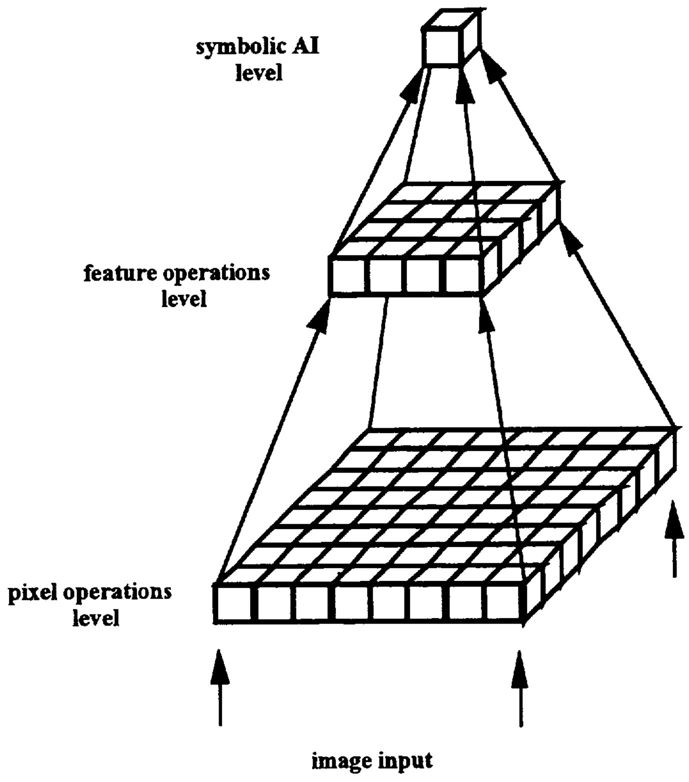

Figure 1-3. Pyramid Image Processor Architecture

discrete processors in a mainframe-like chassis, it is the prototype for future miniaturized integrated imaging sensor-processors used for smart autonomous weapons systems, reconnaissance and surveillance robotic vehicles, and near-earth remote sensing spacecraft. In these advanced imaging sensors, each pixel will have its own signal processing channel, with the pyramidal architecture implemented in a very dense 3-dimensional packaging format directly behind the sensor array.

This kind of hierarchical structure is also proposed for hybrid image-processing computers that make use of combinations of neural networks, fuzzy logic, and symbolic inference or knowledge-base processing. Neural networks can be used at the front end to classify image patterns by partitioning the sample space into fuzzy sets. Symbolic AI and fuzzy logic can then be used to perform operations on the objects defined by the fuzzy sets. This kind of hybrid architecture might have different computer types for each level.

2.3.5 Balanced Systems

Balanced systems are computer systems that are balanced with respect to central processing unit (CPU) performance and throughput. In other words, the system is designed from the start so that the performance of the processing function is matched to the performance of the storage and network functions. System balance is a key requirement for high-performance database systems because these systems have a large amount of on-line storage and a potentially high number of simultaneous users, perhaps distributed over a wide geographic area.

A basic problem that must be solved to achieve a balanced system is closing the I/O gap, the discrepancy between memory and storage I/O rates and CPU performance. Typically, caching methods have been used to alleviate the problem. More recently, microcomputers have begun to adopt mainframe design features such as local high-speed buses. These are short buses designed for high-speed access to random access memory (RAM) and graphics accelerators. The I/O gap also exists between the processor and mass storage. RAM caches are currently used to improve disk access times. Ultimately, new forms of mass storage could lead to further improvements. Examples are the use of larger RAM caches as the cost per bit decreases, use of non-volatile RAM for mass storage, or use of holographic volumetric storage.

The future of balanced systems is most clearly seen in testbed projects at supercomputer centers such as the National Center for Supercomputing Applications (NCSA). Catlett [17] has indicated a need for I/O rates on the order of 128 Mbps with storage requirements approaching a terabyte. With today's supercomputers, the I/O requirements may approach a gigabit (10^9 bits) per second (Gbps). While widespread introduction of synchronous optical network (SONET) and asynchronous transfer mode (ATM) technology should be able to provide ample bandwidth to accommodate the required transmission rates, in both LAN and wide area network (WAN) environments, improvement will be needed in storage technology. Direct connection of the storage system to a high-speed network (called network-attached storage) is one evolutionary improvement. It will eliminate a bottleneck caused by the storage server.

A testbed is being set up at Lawrence Livermore National Laboratory to test improved storage architectures [18, 19]. It is part of the Hierarchical High-Performance Storage System Testbed which employs hierarchical concepts for both storage and networking. Hierarchical storage refers to multiple storage types such as RAM, disk, and tape organized in a hierarchy according to access time and frequency of access. The system will support application data transfers from megabytes to gigabytes per second with total system throughput of gigabytes per second. File size scalability must meet the needs of billions of data sets, each potentially terabytes in size, for total storage capacities in petabytes. The system also is required to scale geographically to support distributed systems with hierarchies of hierarchical storage systems. The circuit switched network will make multiple parallel point-to-point high performance parallel interface (HIPPI) connections between processors and storage devices.

As evidenced in the above discussion, an emerging trend for high-performance networks is the use of parallel networking rather than shared-medium techniques such as Ethernet. In the parallel approach, the full bandwidth of the transmission medium is available to the source and destination by setting up a point-to-point connection. This is basically circuit switching such as is performed in telephone switching systems. The circuit switch allows any source/destination pair to communicate at the full available bandwidth, and for many such pairs to communicate simultaneously. (It does not necessarily allow all pairs to communicate simultaneously, depending on whether the switch has a "blocking" or "non-blocking" architecture.) Another feature of the circuit-switched architecture is that it is well-matched to the capabilities of fiber optics, and is upwardly scalable in performance as photonics technology progresses. A current area of heavily-funded commercial research by AT&T and others is photonic cross-bar switches for direct switching of light beams between input and output optical fibers. These switches will be used in fiber optic WANs but have application to LANs and intra-computer communications as well.

The parallel networking trend complements the trend toward parallel I/O storage drives, such as the disk arrays discussed earlier. These trends can be extrapolated to a system in which parallel disk arrays will be directly connected by 64-bit parallel electronic buses or photonic serial links to parallel processors.

2.3.6 Automated Classification Techniques

In the most basic form of text database, keyword search is speeded up by inverting the data list. An example of an inverted list would be a telephone book re-sorted according to the numerical order of the phone numbers. When databases become more complex or when the volume of new data being added becomes too great, this method may not be efficient. For example, the inverted string lists may take up more memory than the original data. Or, if information is being ingested continuously at a high rate, the inversion process would have to be done continuously, requiring substantial real-time processing resources. A brute-force character-by-character search may then become the preferred method for keyword search. This free-text scan process eliminates the need to invert or to index the database. Since indexing is a very manpower-intensive process, free-text scanning can be cost-effective as the cost of computing goes down. Special-purpose servers and processors are being marketed for this purpose. For example, Benson Computer Research Corp. of McLean, Va., makes a text keyword search scanner that operates at a rate of 1000 pages per second.

Searches based on indexing and Boolean combinations of keywords are often inadequate for precisely focusing a search. AI methods are being applied to this problem, for example in a method known as document clustering for document databases. Document clustering has as its goal the identification of clusters of documents with related contents. Pattern recognition techniques that have been successful in other signature recognition problems can be applied to document clustering by analyzing statistical patterns in document contents. For example, one can generate word and word grouping statistics, and the statistics can be used as signatures for pattern-recognition [20, 21]. However, the ultimate goal of relating documents based on meanings is still beyond the capability of technology, since no one knows how to automate language understanding, nor is there any such method on the horizon.

Image databases can be indexed also, for example by appending date, time, and georeference. However index information is not always useful as a way of searching for the desired information, especially in the case of images, in which information is not encoded in symbolic form. One solution is to automate the process of image classification and to classify according to content features, so that images might be indexed according to category. Since images are non-symbolic information, non-symbolic classification methods are needed. Again, statistical pattern recognition and neural network pattern recognition methods can be used for automated classification. Examples of applying artificial neural networks to automated classification of infrared terrain imagery are described in [22, 23].

3. RESULTS

3.1 IMPACT OF DATABASE TECHNOLOGY ON NASA

Databases for the Earth Observing System will be an important part of NASA information systems in the late 1990s. These databases will be much more difficult to implement than typical databases in terms of the data ingest rate, the amount of data, the geographic distribution of the data, and the number of users. The use of leading-edge technology will be critical to the success of these systems. In particular, NASA distributed databases will have to make use of balanced systems techniques, including a close matching of computer throughput, storage access rates, and network data rates. This implies that the upgrading of networks will have to proceed apace with the design of the other major components.

An example of how balanced techniques may impact system architectures is the use of separate high- and low-speed LANs for data and data management information, respectively, in order to achieve the maximum throughput. In the LAN arena especially, circuit switching seems to be commonly employed for high-speed experimental systems, while ordinary Ethernet is adequate for data management.

Another implication of the developments discussed in section 2 is that data rates on networks may have to go much higher. Even at the present time, the highest data rate service available on the public networks is T3, at 45 Mbps. It is apparent from the projects underway at supercomputer centers that higher rates will be required for the transfer of image data from databases and for motion graphics data visualization. If data are transferred directly to a remote supercomputer, sustained gigabit-per-second rates may be required. This is currently not economical compared to physical transport of magnetic tapes. Nor are these applications sufficient justification for the massive investment in capital equipment that would be required, since supercomputing is still, relatively speaking, a "niche" market. A development that might make gigabit networking economical is the establishment of commercial fiber optics "superhighways" for transmission of video to the home. Such commercial ventures would make it economically justifiable to go beyond 45 Mbps services.

The need for higher data rates will push the development of photonics technology. For short distances parallel-wire systems can be used, but for longer distances serial fiber links are preferred. Electronic-to-photonic conversion circuits currently operate at up to 2.4 Gbps. This may be pushed to 10 Gbps. Eventually, however, multiplexing techniques for fiber will be required because of the limitations of the electronic circuit. The most likely techniques will be wavelength division multiplexing (WDM) and optical frequency division multiplexing (OFDM). These will permit many parallel gigabit-per-second channels to be transmitted over the same single-mode fiber. These techniques have major implications for WAN architectures, and will undoubtedly also greatly impact LAN technology.

The massive amount of funding for photonics research results in the development of new high-speed devices that have bandwidths in excess of electronic component bandwidths, and therefore suggest themselves as alternatives for computing as well as communication applications. As computers evolve toward higher levels of parallelism, inter-processor communication becomes a bottleneck. The photonic switch is an example of a device developed by telephone companies for optical signal switching that has application to inter-processor communication and optical computing. The need to circumvent the limitations of electronic circuitry by eliminating electronic-to-photonic conversions is an incentive to develop all-optical circuitry. The logic of this trend seems to lead in the direction of optical computers. Seen in isolation, optical computers such as the DOC II may not appear competitive with their electronic equivalents. In the context of balanced systems however, optical computing seems to be a logical development for coupling to all-optical networks and storage while minimizing the number of electronic-to-optical conversions.

Associative storage/processing seems to be a common theme of research in the arena of database technology. Again, the watchword is parallelism. Associative systems are parallel systems in which processing and storage are fully integrated and distributed, and could thus be considered an ultimate form of "balanced system". This category includes volumetric holographic storage, optical correlator processors, Hopfield neural networks, and associative array (parallel) processors, all of which are relevant to database processing. Associative memories are theoretically useful for storing images. Digitized images consist of large sets of data words representing pixel values. In the ideal image retrieval system, a small amount of information would be used to recover a large amount – the whole image. Index data could serve this purpose, but does not adequately describe the content. In an associative memory the content description does not have to be a linguistic or symbolic description, but can be simply a subset or compressed representation of the original image. Also, images consist of data from independent, parallel data channels. A lot of image processing occurs at the level of the pixel and its adjacent neighbors. A memory architecture that can store and output data in parallel independent channels would easily couple to a massively parallel pixel processor. Associative memories have this feature. Although these architectures are still in a research phase, we can expect to see these concepts applied to imaging systems and their supporting processing systems in the coming century.

From a systems and life-cycle engineering point of view, the greatest impact to NASA's database projects will occur if there is a breakthrough in storage technology. It is extremely difficult to change the storage technology during the life cycle of a system, since the existing accumulated data must be transferred to the new storage medium. If the system was designed to handle a real-time ingest rate, the new system would have to handle a greater-than-real-time rate just in order to transfer the old data in a reasonable amount of time. Therefore any upgrade would require a capability for a several-fold improvement in ingest rate. This creates an incentive for pushing a new technology to the earliest possible deployment.

3.2 TECHNOLOGY ROADMAP

A preliminary technology roadmap, or possible sequence of database technology insertion, is presented below.

1996-2000: The first petabyte wide-area distributed databases may be developed by NASA. Methods of improving system balance will be put into practice in these database systems. This may include RAID disk arrays, large RAM caches, high speed disk interfaces, network-attached storage, and circuit-switched LANs, such as HIPPI and FCS. Magnetic tape will probably continue to be the preferred medium for archiving and backup of the data. Storage devices with more advanced parallel access capabilities may be employed, such as optical disks with multiple heads or holographic parallel read-out. Parallel computers will be used for database applications, and architectures optimized for database retrieval, such as associative SIMD array processors, should become more widely used.

2000-2010: Economies of scale will bring down the cost of gigabit per second networking, so that NASA databases will be accessible nationwide over the commercial networks at reasonable cost. ATM and SONET will provide bandwidth-on-demand service up to gigabits per second. Three-dimensional holographic storage will begin to appear commercially. Non-volatile RAM may also be used for mass storage. Parallel computers for database processing will use chips with multiple processors per chip, and will make use of associative architectures. Computers will make use of quantum-well semiconductors, molecular switching devices, and/or photonic components to achieve high density, speed, and connectivity. Imaging instruments as well as computers may be constructed in three-dimensional parallel architectures employing photonic signals to achieve high inter-channel connectivity within the three-dimensional circuit structure.

2010-2020: Associative memories based on the Hopfield neural network may become available. Advances in all-optical and molecular computing will be used for database applications.

As second- or third-generation AI products begin to show "human-like" information processing capabilities, these capabilities will be applied to finding and retrieving data, thereby improving the efficiency of database searching and increasing the relevance of the information presented to the user. Developments in molecular level storage technologies will greatly increase the ability of database systems to efficiently store, manipulate, and search through huge amounts of data.

3.3 FEASIBILITY AND RISK

Parallel storage: Parallel storage systems such as RAID are essentially off-the-shelf, but are dependent on the underlying magnetic disk technology. This is essentially an intermediate milestone on the road to more advanced systems, and is low-risk. Holographic storage is under development and may be brought to market by the year 2000, although efforts to date have fallen short of expectations. This technology should be thoroughly evaluated as a potential alternative to magnetic disk systems.

Associative memory: Associative memory using Hopfield neural network and inner-product (correlation) approaches is one of the few architectures that has the potential to make use of parallel processing and memory technology beyond the density and scaling limits of conventional transistor-based VLSI. Its importance therefore goes beyond database applications. A great deal of research is being devoted to Hopfield associative memories, but the method appears to need more research before it can be evaluated for practical devices. Research appears to be stuck in the proof-of-concept phase. This may be because semiconductor device technology has not yet reached the stage to take full advantage of the method.

Optical database computers: Database applications are a sufficiently general problem, and yet have sufficiently specialized computational aspects that they are a good candidate for new computer architectures tailored to databases. For example, a free-text search processor is basically a digital correlator. Currently, correlator processors are probably best built using VLSI correlator chips, but eventually the high parallelism of optical computers may give them the edge. The work being carried out by Peter Guilfoyle on the DOC III is state-of-the-art for complete, working optical computers. To leapfrog that research effort would require breakthroughs in device technology, specifically mainly in the area of ferroelectric liquid-crystal or other spatial light modulators.

Pyramid and hierarchical architectures: This is primarily an architectural approach for parallel computers, storage, and networks, and is therefore a near-term, low-risk developmental area. Pyramid architectures implemented in miniature packages for space- and airborne applications are a longer-term development.

Balanced systems: This is a related architectural methodology that may include parallel and hierarchical approaches, and is a near-term, low-risk, though perhaps expensive approach to improving database system throughput.

AI systems: AI technology, including both symbolic AI (expert system), fuzzy logic, and neural networks to text and image database search is becoming available in the form of off-the-shelf commercial products. The current products are only first steps, and it is unclear how useful they are. There is still a great deal of room for improvement, for example, in finding new methods of generating signatures that allow documents to be compared. It should also be kept in mind that like all AI-base products, these systems have little to do with intelligence and content-understanding, but are actually more like sophisticated filter algorithms that operate on external signatures of meaning, not the meaning itself. The goal of intelligent search in the human sense is therefore not yet on the horizon.

Perhaps the highest payoff in database applications will come from breakthroughs in storage technology, but staking a design on such a breakthrough is risky. For example, Tamarack Storage Devices planned to market a holographic storage system in late 1993 or early 1994, but was unsuccessful. Another perhaps less risky storage breakthrough will be the development of semiconductor RAM devices that are non-volatile and have sufficiently high density to compete with magnetic disks for random-access random storage. It seems certain that NASA needs to make a commitment to these key technologies perhaps by supporting risk-reduction programs. Risk reduction will be important to accelerate deployment and to minimize the risk that the NASA databases will have to be transferred to a new medium.

References

1. Wilke, J.R.,"Supercomputers Manage Holiday Stock", *Wall Street Journal*, Dec. 23, 1992, p. B1.

2. Romine, C.L., "Fiber Unleashes High-End Visualization Power", *Photonics Spectra*, February 1993, pp. 77-80.

3. Berra, P., et al., "The Impact of Optics on Data and Knowledge Base Systems", *IEEE Trans. Knowledge and Data Engineering*, March 1989, p. 111.

4. "Tamarack's $22M Consortium for Wave Guide Holo Storage," *Holography News*, April/May 1995.

5. Gallant, J., *EDN*, April 9, 1992, p. 61.

6. Grosspietsch, K.E., "Intelligent Systems by Means of Associative Processing", in *Fuzzy, Holographic and Parallel Intelligence*, Branko Soucek, ed., New York: Wiley, 1992.

7. Hopfield, J.J., and D.W. Tank, "Computing with Neural Circuits: a Model" *Science* , 233, 1986, pp. 625-633.

8. Abu-Mostafa, Y.S. and J.M. St. Jacques, "Information Capacity of the Hopfield Model", *IEEE Trans. on Info. Theory*, 31(4), July 1985, pp. 461-64.

9. McEliece, R.J., et al., "The Capacity of the Hopfield Associative Memory", *IEEE Trans. on Information Theory* ,33(4), July 1987, pp. 461-82.

10. Hua-Kuang Liu, "Optoelectronic Inner-Product Associative Memory", *NASA Tech Briefs*, March 1993, pp. 45-6.

11. Lambe, J., et al.,"Electronic Neural Networks", *NASA Tech Briefs*, January 1988, pp. 42-44.

12. Shimazu, M., "Optical Computing Comes of Age", *Photonics Spectra*, November 1992, pp. 66-76.

13. Guilfoyle, P.S., "A Third-Generation Digital Optical Computer", *Photonics Spectra*, May 1993, pp. 116-124.

14. Abu-Mostafa, Y.S. and D. Psaltis, "Optical Neural Computers", March 1987, pp. 88-95.

15. Farhat, N., et al. "Optical Implementation of the Hopfield Model", Applied Optics, Vol. 24, No. 10, May 15, 1985.

16. Potter, J.L. and W.C. Meilander, "Array Processor Supercomputers", *Proc. IEEE*, Vol. 77, No. 12, December 1989, pp. 1896-1914.

17. Catlett, C.E., "Balancing resources", *IEEE Spectrum*, September 1992, pp 48-55.

18. Mencke, S.M., “Livermore Hopes to Speed Transfer of Huge Data Sets”, *Government Computer News*, June 8, 1992.

19. "High Performance Storage System Project of the NSL", WWW page at http://www.llnl.gov/liv_comp/nsl/hpss/hpss.html.

20. Busch, E., “Search and Retrieval”, *Byte Magazine*, June 1992, pp. 271-276.

21. Combs, N., “Visualizing the Semantic Content of Large Text Databases Using Text Maps”, *1993 Goddard Conference on Space Applications of Artificial Intelligence*, pp. 249-261.

22. Yates, G.L., and S.J. Eberlein, “Hierarchical Pattern Classifier”, *NASA Tech Briefs*, June 1992, p. 84.

23. Chettri, S.R., “The Probabilistic Neural Network Architecture for High-Speed Classification of Remotely Sensed Imagery”, *1993 Goddard Conference on Space Applications of Artificial Intelligence*, pp. 119-132.

CHAPTER 2. SOFTWARE TECHNOLOGY TRENDS

SUMMARY

Increasing the efficiency of the software production process has proven to be a formidable task. Over the years, many different approaches have been tried. Among those discussed here are use of computer-aided software engineering tools, object-oriented programming, fourth generation languages, and software reuse. Although these technologies can produce productivity increases, potentially as high as 10-to-1, most are still in their infancy. Most will also require substantial investment in retraining and reorganization, including such new approaches as software reuse libraries. This investment will typically be over a number of years, and substantial return on investment will only appear after a correspondingly long period.

The widespread trend toward a client-server computing environment, and the attendant need to deal with the many legacy systems left in the wake of this transition, require that efforts to increase the productivity of software production be continued and even expanded. The realization of truly portable software that is fully interoperable, independent of the tools with which it was developed and the platforms on which it runs, will depend on the establishment and widespread adoption of industry standards. The convergence of software development tools, object-oriented technology, and fourth-generation programming languages also has the potential to assist in achieving this ambitious goal.

1. INTRODUCTION

During the 1980s, a great deal of effort was devoted to devising new software techniques and methods to cope with the perennial problem of the high cost of software development. These methods promised great improvements in software productivity, as high as ten-to-one. However, the proliferation of these new technologies has led to a certain amount of hesitancy on the part of managers in adopting them, since there are no clear guidelines as to how one might select among them, what the cost will be, or how long it might take to see a return on investment. At this point, this hesitancy is probably justified, since many of the new software technologies are still in their infancy. It is clear, however, that organizations will have to adopt methods for improving software productivity and reliability in order to remain competitive.

2. LONG-RANGE TECHNOLOGY TRENDS

2.1 ANALYSIS OF TRENDS

2.1.1 Computer-Aided Software Engineering (CASE)

Computer-aided software engineering (CASE) tools have not been widely successful to date because of the existence of "traditional" application development methods, as well as the lack of standards for fully integrated toolsets. CASE was intended to improve applications programming productivity by automating the more tedious aspects of software design, but productivity has actually increased very little. The quality of applications has improved from using CASE tools, but the success of CASE depends upon increasing productivity as well. The three general categories of tasks in programming are enabling creativity for analysis, recording modeling data, and transforming models into other representations. Ideally, CASE tools should support the initial creative operation and allow ideas to be shared among software development team members. Also, the tools must be able to transform models from analysis through design to construction.

CASE has varying definitions within the industry. Some claim it completely automates program development and replaces existing methods; others claim it includes any new applications development technologies and can co-exist with existing methods. Some feel that CASE is outdated and will be replaced by other programming methodologies; others contend that CASE, while still in its infancy, has the potential to produce tremendous rewards in the future [1].

Whereas the traditional programming methods are process-driven, which means that the programmer first designs a screen then creates a file and generates a program, CASE is data-driven. With CASE, the programmer starts with the database through which the system is prototyped utilizing concepts such as interactive group design and joint application. Through interactive group design, the development team and the end-users work together in a much more iterative fashion, resulting in a more accurate application design. With joint applications, groups of applications that have like data can be modelled concurrently. The overall model basically represents logical groupings of application components.

Ideally, the significance of this design approach is that when application design changes arise, they can be accommodated easily at the data model level and propagated through the system as opposed to being done at the code level, which results in inaccuracies, inefficiency, and redundancy. The entire process of changing and/or adding applications becomes more rapid, error free, and far less expensive. At present, this scenario is more the ideal than the actual practice.

So far, CASE tools have failed to work effectively on a wide scale. Leaving aside the consideration that most corporations are not willing to write off all existing software development approaches as failures and start anew with CASE, the blame seems to fall on vendors, users, and the very nature of CASE, in no particular order and with considerable overlap. Developers are not willing to pay the large price for switching to the CASE programming environment; the tools are expensive and the learning curve is steep. Most developers that choose to implement CASE do not invest the required resources, get frustrated, and resort to their old methods of programming.

The situation is beginning to change, however. The recent trend toward client-server architectures has been a major factor contributing to a resurgence of interest in CASE tools – tools that are less expensive and easier to work with – this time for the PC environment. The newer CASE tools tend to deal with only a single aspect of software development in contrast to the older integrated CASE tools that treated the entire software development process, were massive in size, and usually employed proprietary approaches that precluded interoperability with software developed using other manufacturer's tools [1]. CASE vendors are starting to work both with other vendors and their customers. Application developers are becoming more willing to use CASE tools because the vendors are promising to incorporate their suggestions into the future releases of the CASE tools. In the future, standards will be implemented allowing for data sharing among CASE tools, development of client-server architectures, and development of applications across multiple platforms. Developers will only have to write applications once, and CASE tools will easily produce applications for mainframe, workstation, client-server and/or cooperative-processing environments.

The new CASE tools incorporate new features such as object-oriented methodology and support for graphical user interfaces (GUIs). Because of their smaller size, the new tools are a lot less expensive than their predecessors and are easier to learn and use. Renewed interest in CASE tools also has been aided by the increased use of object-oriented programming (OOP) and fourth-generation languages (4GLs), such as PowerBuilder, SQL Windows, and Visual Basic [2]. Use of these languages for large projects requires additional software development support that can be provided by CASE tools. There is a synergy between CASE tools, OOP, and 4GLs. CASE allows a program to be built from easily changed modules which are characteristic of OOP. While CASE vendors are building into their products bridges to 4GLs, many 4GL developers are also building bridges from their products to popular CASE tools.

Whereas the older integrated CASE tools were produced mainly by large, established companies, the newer PC-based versions are being developed by new, small companies such as LBMS, Popkin Software and Systems, Cognos, Gupta, and Sybase. The products produced by these companies are flexible, open, and distributed. As a result, these newer tools are becoming one of the hottest technologies of the 1990s while the older integrated CASE tools are on the decline.

2.1.2 Fourth Generation Languages

The 4GLs are powerful application-development tools that are easy to learn and greatly reduce the amount of code required by programmers to design, write, and test applications. As a result of using 4GLs, developers have reduced programming costs and have increased productivity by 10-to-1 or more over lower level languages. By using 4GL tools, programmers can more easily manage software projects, rapidly prototype applications, more easily maintain and modify software applications, and write more consistent software documentation.

As part of the rapid evolution of the computer industry, several generations of software development tools have emerged. Using first-generation languages, programmers communicated with the computer in machine language, a series of bits in the binary base. First generation programs proved to be very difficult to write, test, and maintain. In second-generation languages, productivity increased through the use of mnemonic assembler languages, which allowed programmers to specify an instruction and have the assembler translate it to binary code. Third generation languages (3GLs), which are the most widely used languages for programming business applications, use an English-language construct. Languages such as BASIC, COBOL, C, and FORTRAN are procedural-based 3GLs. Fourth generation languages offer another level of productivity by allowing users to specify what to do instead of how to do it.

While there is no fixed definition of a 4GL, the following attributes are considered in determining whether a tool is 4GL-like:

- Prototyping – 4GLs support a working model to develop applications, allowing direct communications between users and developers. Users often can see immediate results, and the result of this process may be retained in the final production system.
- Concise Code – A 4GL has to do more with fewer lines of code. A common metric is that a 4GL has the ability to perform a task with one-tenth the number of lines of code as COBOL.
- Portability – 4GLs often offer hardware independence. 4GL-developed applications may often be ported freely from platform to platform.
- Data Dictionary or Repository – These central warehouses of information on data structures often are integrated into 4GLs.
- Intelligent Defaults – 4GLs save time by giving programmers a menu of options and, wherever possible, offering assumptions about the most common option or parameter. The programmer then can concentrate on the exceptions.
- Application Generators – For common routines, such as reports and file maintenance, application generators can be used to produce the finished product, relieving the programmer of the tedious work of repeatedly producing the same type of program with minor differences.

Initially, 4GLs were not widely accepted because employing these toolsets required large hardware resources, and a steep learning curve discouraged organizations from adequately retraining programmers. However, the advantages that 4GLs offer, such as portability, prototyping, intelligent defaults, and concise code, outweigh these drawbacks. Some of the concerns about 4GLs have been alleviated by vendors who have ported their products to machines which allow users to take advantage of open systems environments and client-server architectures.

The downsizing of mainframe-based applications to distributed networks of minicomputers and PCs has multiplied the role for 4GLs. Indeed, the manufacturers of some 4GLs tout their ability to help developers migrate mission-critical systems off legacy systems incrementally. There is a new class of users interested in developing ad hoc applications or querying databases, roles perfect for the latest generation of 4GLs. The features of the open systems environment and client-server architectures are congruent with the aims and capabilities of 4GLs.

Recently, several of the major 4GL vendors have enhanced their products to obtain a larger segment of the client-server application development market. Enhancements include broad support for a range of hardware platforms and databases, provided by a wide range of database- and hardware-specific drivers. OOP technology, visual programming (e.g., Visual Basic), or a combination of the two characterize some of the offerings. Users in a client-server environment want quick response times while making efficient use of system resources. Accordingly, a number of 4GL vendors have modified their products to be more efficient in the way code is compiled and in the way data elements are defined, accessed, and maintained in the development of database tools [3].

The future of 4GLs is assured because of their relationship with other long term software methodologies, including database management systems, OOP, and CASE tools. At least one vendor provides compatibility with 16 different CASE tools. OOP is expected to become more important in building graphical front-ends with 4GLs. Whether 4GLs remain stand-alone entities or become inextricably integrated with CASE technologies, they will continue to play a role in applications development. Programmers increasingly need "industrial-strength" capabilities to build mission-critical applications independent of toolsets. 4GLs will continue to be at the heart of any complete application solution.

2.1.3 Object-Oriented Programming

OOP helps enhance individual programmers' productivity, shortens development life cycle, facilitates group development projects, and streamlines program maintenance. Objects are self-contained software modules that perform a given set of tasks on command. By connecting a number of these stand-alone objects, a software engineer can easily create a complete, working application within a short period of time. The difficulty arises in creating an object that works properly and is robust enough to be used in a variety of applications. However, once a given object exists in finished, working form, other programmers need know nothing about how it accomplishes its task. Since objects in OOP reflect real-world entities, they can more closely reflect the way an organization conducts business than would programs based on abstract data types. OOP is ideal for large-scale team software developments because the objects always function the same way, regardless of the applications in which they have been used.

Relying on modular software units that can easily be connected, reused, and enhanced, the object-oriented approach boosts the productivity of individual programmers, simplifies team development efforts, and streamlines maintenance down the line. Because object-oriented programs are inherently better organized and more stable, OOP permits larger, more complex applications than are practical with traditional procedural programming methods. And with the proliferation of object-oriented software environments, object-based design makes it possible to integrate applications on a system-wide level.

The most commonly used object-oriented programming language is C++, a modified version of the widely-used C language. Because of this association with C, developers can ease the migration process to the OOP environment. While initial development efforts in C++ offers some advantages over procedural languages, the real payoff for OOP development comes when programs have to be maintained, expanded, or ported to other platforms. The increased importance of cross-platform development in a networked setting also provides a strong impetus for moving toward OOP. When an application outgrows the machine for which it was designed, an almost inevitable outcome, being able to port functionality or entire programs quickly and easily from one environment to the next takes on a high priority.

The decision to switch to developing applications using object-oriented technology can be very difficult. There are still many software programmers who were trained in developing applications using procedural-based languages, and are unfamiliar with the structures of the OOP paradigm – including objects, classes, polymorphism, and inheritance. Retraining programmers in OOP can take, on average, six months, which can be very costly and impact current development schedules. Another concern for switching to OOP is that object-oriented development tools are still in short supply for many of the hardware operating system platforms.

Paramount in any decision to move toward object technology is the hardware plan. Corporations that have decided to retain a traditional computing strategy centered around minicomputers or mainframes may not find the case for object-oriented development very compelling for the time being. GUIs, with all their programming complexities, are not currently an issue on mainframes. The expanding universe of GUIs on small computers, however, encourages the move to OOP for these machines. While the programming requirements for most GUIs do not meet the strict definition for object orientation, many aspects of the object-oriented paradigm are applicable. This is especially true of the user interface itself, with all of its familiar objects, including windows, dialog boxes, menus, and icons. Object-oriented languages are a natural for customizing these essential interface elements.

As OOP technology becomes more widespread, third-party objects and object libraries will become available for use as building blocks for new systems, effectively reducing the overall development costs. Because these object libraries will be available for general use, they will be thoroughly tested and robust, thus reducing the effort for product testing. Presently, object-oriented technology has not been fully embraced by corporate development teams. Although the payoffs for using OOP techniques appear to be great over the long run, day-to-day business pressures prevent corporations from changing their development practices.

Despite the difficulties inherent in converting to a new technology, most of the software industry today sees the object-oriented approach as the preferred direction for future software development [4]. By its nature, OOP lends itself to development of reusable software components which will be essential to increase software productivity. To build such reusable components, however, it will be necessary to have standards in place to ensure that components built at one time and place will mesh properly with those built elsewhere and perhaps at different times. The Object Management Group (OMG) is a key industry group that is writing object standards. In 1994, the group released version 2.0 of its common object request broker architecture (Corba), which defines the mechanisms by which objects interact, to provide a means of making object names globally compatible. OMG has also developed other standards covering the naming of objects, relationships and transactions among objects, and the building of objects for application-related functions occurring in distributed system applications.

The initial driver for the development of object-oriented components is seen to be various vertical markets such as the insurance and communications industries. These needs will provide an opportunity for small independent software vendors who are familiar with the particular markets, making them competitive in these areas with the software giants, such as Microsoft, Lotus, and Computer Associates. Systems integrators will also be players as they develop solutions for their clients. Systemshouse SHL and Andersen Consulting have projects underway to develop reusable objects for use in vertical markets of interest in the insurance, banking, and telecommunications industries. These objects are being developed internally or bought from independent software vendors [5].

Most corporations are actively investigating object-oriented technology; these efforts will include many strategic pilot applications. Microsoft is developing a cross-platform implementation of its Object Linking and Embedding technology. Large-scale object-oriented applications are being developed, with a significant trend toward platform-independent object-oriented tool kits. Platform-independent object-oriented development environments will be established as the preferred strategy for most corporate development teams. Microsoft is leading the way in this area by putting object technology to use in its Visual C++ and Visual Basic developer's tool kits.

2.1.4 Interoperability

Several efforts are under way which are intended to encourage the development of software that can operate on multiple hardware platforms. This process is known as interoperability. Such a hardware-independent design would reduce software development costs and provide a broader customer base for applications. Also, software maintenance and enhancement efforts would be reduced because only one platform would need to be addressed. Software developers, as well as customers, would greatly benefit from interoperability.

The advantages gained with interoperability are numerous. For developers, the advantages of such a technology are clear: reduced development, production, and distribution costs; a wider customer base; and ease of application maintenance and enhancement. Developing one version of an application that can run on any platform obviously costs much less and requires far fewer resources than developing a separate version for every supported platform. Production and distribution of a single product would greatly simplify tracking and inventory operations.

Most important for developers, however, would be the access to new markets that hardware-neutral software would provide. If a developer's application is capable of running on every hardware platform, the list of potential customers suddenly expands from users of a few platforms to users of most or all computer systems. Conversely, competing developers also would have access to broader markets and those developers who have captured platform-specific markets could lose the protection afforded by developing for a niche platform. However, the benefits gained from exposure to the new customers could outweigh the risk of increased competition.

For the user, the benefits of interoperability are equally significant. First, the cost of applications would decrease due to two factors: reduced development and distribution costs and increased competition from a larger base of developers. Second, and most important, the user would gain enhanced flexibility in both business operations and hardware purchasing decisions. Applications would run on almost all existing hardware platforms providing for greater intercommunication among users. Also, the user would experience an increase in flexibility for hardware purchases because the user could purchase the system which offers the best price/performance specifications.

One effort which is striving to make interoperability viable is the Open Software Foundation (OSF) Architecture-Neutral Distribution Format (ANDF). OSF sees ANDF as the key to true interoperability and is developing a two-part technology that consists of a standard application development environment and a hardware-specific installation mechanism to facilitate the process. Products can be developed for this technology, but real commercial development lies in the future; there are still obstacles to overcome, including the instinct to compete. ANDF can be a reality only when the industry decides it can benefit more from cooperation than competition.

2.2 FUTURE APPLICATIONS

2.2.1 Rightsizing

It will become increasingly important to ensure compatibility of applications and the platforms on which they are installed. Rightsizing is the process of moving applications from their current locations to platforms where they can run more efficiently. This process involves downsizing, moving from a minicomputer or mainframe platform to a microcomputer platform, or upsizing, moving from a platform that the application has outgrown. Another definition of downsizing is the practice of moving applications closer to the end-user; this is the rationale behind terminals replacing punched cards, as well as client-server architectures replacing mainframes, which is presently the most common form of downsizing. Computer users, managers, and application developers will spend much of the 1990s addressing the rightsizing needs of organizations large and small.

Today's open systems, particularly UNIX reduced instruction set computer (RISC) systems, can provide performance and value superior to that of mainframes by factors up to six-to-one. In addition, UNIX RISC systems are improving their price/performance each year by almost 50%, while mainframes improve by only about 10%. Therefore, the gap between RISC system and mainframe performance continues to widen. Today's six-to-one RISC system price/performance edge over mainframes will be next year's ten-to-one edge.

Essentially the same trend drives the evolution of software and peripherals, where open systems also are overtaking mainframes: UNIX high-volume on-line transaction processing (OLTP) and data center applications are currently run only by leading edge users. UNIX throughput is advancing so fast that it is starting to be widely used for high-volume OLTP. Within the next few years, UNIX will become the mainstream option for OLTP and data center applications. Also, both relational and object-oriented database management system (DBMS) products are much more widely available on UNIX open systems platforms than on mainframes. Another advantage for UNIX systems is that their large disk subsystems have nearly a ten-to-one cost advantage over their mainframe-based counterparts.

Two strong industry trends have been the driving forces behind these current trends: the increasing power of commodity microprocessors; and, the computer industry's move to open systems for solving large-scale business computing problems. For these reasons, many companies are downsizing their computer resources from mainframes to client-server architectures. Some 80 percent of 75 companies interviewed by Forrester Research report they are increasing downsizing efforts. More than 70 percent of 300+ U.S. companies surveyed recently by Dataquest are investigating or have already off-loaded software from mainframes, and more than half regard their mainframe as a database or file server.

Rightsizing will produce more and more client-server architectures, with these networked systems consolidating computer resources and applications. Fewer developers will be required to generate applications. More software packages will be available to the end-user in real time. Fewer applications will reside on mainframes. Rightsizing will require programmers to develop future applications with consideration toward scalability; applications will be designed to run on a small platform, but easily movable to a larger platform as the database size increases, without affecting functionality or requiring rewrites of the application.

As the 1990s progress, rightsizing promises further productivity improvements as more and more mission-critical applications are implemented on PCs connected by local-area networks (LANs), as application portability across multiple platforms becomes easier, and as object-oriented programming techniques enjoy increasingly widespread use.

2.2.2 Legacy Code

Dealing with applications that are hosted on platforms that are part of outdated system architectures is a widespread and thorny problem. Many organizations have invested heavily to develop their current software systems. Many have expended great amounts of time and money in employee training, software development tools, and hardware development suites. With these investments, programmers have developed many applications using the past and current technologies. However, as companies start to use the latest software development processes or switch from mainframes to client-server architectures, they will be unable to easily integrate their previously developed software modules into these new environments. This software is known as legacy code.

Ideally, developers would like to be able to easily incorporate previously developed software into their new applications. By utilizing software from their past development efforts, programmers would be able to reduce redundant programming efforts, as well as to maintain a common "look-and-feel" among their applications. Presently, no universal method for porting legacy code to varying hardware platforms exists. Individual, time consuming efforts must be performed for porting legacy code to each platform. The future usefulness of legacy code is dependent upon the development of systems that can easily port this software to a multitude of development environments and hardware platforms. Current efforts in this direction include the creation of software reuse libraries for object-oriented code and the development of software reverse-engineering and re-engineering tools.

2.3 EMERGING TECHNOLOGIES

2.3.1 Software Reuse Libraries

Software reuse libraries offer many benefits and will continue to grow as effective programming resources for increasing programmer productivity. Over the next decade, the use of software reuse libraries may become the biggest breakthrough in increasing programmer productivity. By reusing software designs and models, programmers can greatly reduce software development time and costs. Ideally, future reuse libraries will contain vast amounts of software modules whose functions are tailored for specific operations on a variety of hardware platforms. These reuse libraries would support high levels of reuse and double or triple effective programmer productivity.

While the concept of software reuse libraries is highly regarded, the general practice of using these libraries is still in the beginning stages for software developers. Some of the main drawbacks of software reuse libraries include the following issues. Most programmers prefer to design and build software from scratch over reusing existing software because they believe they can write software which is tighter, faster, or more elegant than the existing software. Also, the exact function required by a module may not exist in the available reuse library; thus, the programmer would be required to design and develop the module from scratch. Programmers fear that reuse libraries will remove the creativity in software development. However, software development must now be regarded as a science as opposed to an art form in order to leverage gains from previous software development efforts into future projects. Programmers must start to develop software with reuse in mind.

Reuse on a measurable scale starts with reusable component libraries, but it is uncertain how the component libraries will be created, as well as maintained. Software donated to reuse libraries must be written with a very high degree of precision; the persons maintaining the software must be able to understand the code in case changes to the software are required. In the past, programmers have been unwilling to donate their software to reuse libraries because they either want to get paid or want to be assured they will not be responsible for updates or maintenance. In response to this problem, some companies have formed groups of full-time developers for creating components for inclusion in software reuse libraries.

In order to assure the success and effectiveness of software reuse libraries, users must obtain code from these libraries that meets their requirements and performs the specified functions without any need for customization or correction. Also, reuse libraries must be widely available and easily accessible to programmers. The success of software reuse libraries as effective resources for increasing productivity depends greatly upon their acceptance by programmers.

2.3.2 Total Quality Management (TQM) and Metrics

TQM is a method in which the development process is continuously monitored and corrected in order to produce the highest quality products; solutions to problems are continuously incorporated into the development process. Most information system (IS) organizations have implemented basic quality tracking programs in which fundamental tests are performed; however, little progress has been made in modifying their development processes as a result of these quality programs. While IS organizations may use TQM in the early stages of program development, fewer than five percent of IS organizations keep up quality improvement throughout the product life cycle. Although software developers are pressured to build a new generation of graphical and distributed applications, little commitment is made to develop quality improvement programs for streamlining the development effort. The success of TQM efforts will be achieved only through firm commitment by management. Tests, metrics, and quality measurement tools can be developed to assess the quality baseline and to measure improvements over a period of time.

Along with increasing pressure to deliver better quality products in shorter time frames and for less money, IS organizations are having to develop applications utilizing the latest technologies: client-server architectures and object-oriented technology. These new development paradigms are impacting quality improvement and measurement methodologies, as well as productivity. The increased demand for highly complex systems has put a great burden upon software developers. Although developers have increased their productivity through the use of advanced development tools, the increased demand for complex systems have masked their growth in productivity.

The main problem with implementing TQM programs is that change is required, which can be very painful and lengthy. To effectively implement TQM into the development process, management must develop a comprehensive implementation plan, which may call for a ten year effort to completely incorporate TQM into the development process. Management is usually reluctant to sign up to any plan which has a great length of time before a return on their investment can be realized. The easiest way to convince management to implement TQM programs is to determine the cost breakdown for development efforts, which include: effort for initial development, detecting errors, and repairing errors. This data is not hard to track, and in most cases, is already available through project management systems that track walk-throughs, reviews, and defect rates. An average IS organization can spend from 40 percent to 50 percent of the their budget on fixing defects caused by poor quality.

To combat behaviors such as the "code or die" syndrome that seem to sabotage the drive toward quality, Coopers & Lybrand modified their four-phase TQM methodology -- assessment, planning, process improvement and integration -- to suit the tenets of software engineering. The centerpiece of the assessment phase is to develop metrics that assess the quality baseline, and to measure improvements over time. These metrics should be tailored specifically to the key quality issues of each organization. NASA has developed a method for determining the appropriate quality issues to be measured. This technique is called goal-question-metric (GQM), and is used to refine individual components of key quality issues, and ultimately the metrics that might be derived from them. The second phase, planning, is based on determining the direction which the organization wants to pursue by incorporating the highest-priority quality issues. The "plan" developed is actually a short list of things to do over the next six months, and contains the greatest near-term opportunities to increase quality. Process improvement, the third phase, determines, through the use of metrics, which areas of the development process require refinement and what modifications are required. Finally, the integration phase is used to implement changes to the development process.

As mainframe-based applications shift to distributed architectures, new development methodologies are required to handle the added complexity. For instance, CASE tools can help to mitigate the extra layer of complexity added by the new development techniques of GUIs and client-server architectures. The use of these new complex development tools has made obsolete traditional software metrics, such as lines of code. In the future, new software metrics and productivity measurement tools will need to be developed in conjunction with the new programming methodologies.

3. RESULTS

3.1 IMPACT OF SOFTWARE TECHNOLOGY ON NASA

NASA's OSC will need to develop a strategic plan which should include an optimum migration path for integrating the latest and projected software development technologies into their current development processes. The strategic plan should include a study of NASA's current development processes, the latest software development technologies, integration methodologies for inserting these new technologies, projected productivity improvements and returns on investment, impact on organization and management structures, and education and retraining efforts for the development teams.

As a result of using these advanced programming techniques, developers have reduced programming costs and have increased productivity by 10-to-1 or more over traditional methods. However, over the next decade, the use of software reuse libraries may become the biggest breakthrough in increasing programmer productivity.

Programmers will require retraining on most of the new software development technologies. Retraining programmers to become proficient in these new technologies can take, on average, six months, which can be very costly and impact current development schedules. Also, restructuring the software development organization can take several years to complete.

3.2 TECHNOLOGY ROADMAP

A technology roadmap, or hypothetical timeline, of the development of software technology over the next 25 years follows.

1996 - 2000: Companies will develop large-scale object-oriented applications and test them using strategic pilot studies. There will be increased acceptance of client-server CASE tools and further development of CASE standards. The first software re-use libraries will be established.

2000 - 2010: CASE, 4GLs, and OOP will merge to form a powerful, efficient software development environment. Reusable software modules will be increasingly employed. Software will be more interoperable with other software and platform-independent. Open system environments will become increasingly common.

2010 - 2020: Client-server architectures will be pervasive in an open-systems environment. Software will be fully portable with modules transferable among all platforms. Widespread use of CASE, 4GLs, and OOP will result in mostly machine-generated reusable software.

3.3 FEASIBILITY AND RISK

NASA's OSC should be major participants in the current drive to improve software productivity. Most of the software development technologies discussed are still in their infancy. While the current potential of these new technologies may not be fully utilized presently, they still require more improvements to gain the strong support of most developers and to revolutionize the entire software development process. The industry will be driving strongly over the next decade toward developing an open systems environment which will help to further advance the capabilities of these, as well as projected software development technologies.

Some studies have been made to determine when the advantages of introducing new methodologies outweigh the disadvantages, but managers are wary of relying on such studies, because the number of variables is large, and they are hard to control. More data will need to be accumulated, but it seems clear that benefits will typically be slow to materialize. Several years of retraining and organizational restructuring might be required, making it hard to justify investment on a yearly basis. This fact will complicate the budget-justification process for software development projects within NASA. Methods will need to be developed for justifying investments over a multi-year period to achieve the long-term benefits. Areas which might show relatively quick benefits and are fairly well-proven include graphical-oriented GUI development tools, and 4GLs. Methods requiring long-term investment include OOP, software reuse libraries, and CASE.

NASA can expect that its software contractors will be forced by competitive pressures to adopt the new methods, since it is clear that traditional software development methods will no longer be affordable in the future. Besides a reduction in the cost of software development, NASA can expect to see other major benefits coming from the new technologies, specifically, a substantial reduction in maintenance costs and a substantial increase in software reliability.

References

1. Rettig, H., "Case Study: The Next Generation," *VAR Business*, February 1, 1994, page 87.

2. Moran, R. and J. Soat, "CASE Gets Personal," *InformationWeek*, June 27, 1994, page 40.

3. Bowen, B.D., "Vendors Boost Their Product Offerings -- 4GL Tool Makers Enhance Their Wares to Attract Client-Server App Developers," *Open Systems Today*, February 7, 1994, page 3.

4. R. Comerford, "Software Engineering," *IEEE Spectrum*, January 1995, p. 62.

5. P.J. Gill, "Knit 1 Pearl 0 -- The Rise of Modular Software Components, Based on Object-Oriented Technology, Will Redefine both the Software Industry and the Nature of Software Design Itself", *OEM*, May 1, 1995, page 64.

Bibliography

Major, Michael J., “The unresolved case: CASE promises great productivity, but users still are awaiting the evidence.”, *MIDRANGE Systems*, April 28, 1992, Vol. 5, No. 8, p. 25.

Krivda, Cheryl D., “Start your engines; application development is driving toward client-server environments.”, *LAN Computing*, Dec 1991, Vol. 2, No. 18, p. 18.

Bird, Chris, “CASE crop a flop.”, *Software Magazine*, Nov 15, 1991, Vol. 11, No. 14, p. 8.

Stodder, David, “The gospel of efficiency: tool integration and productivity: CASE goals for the 1990s.”, *Database Programming & Design*, Oct 1991, Vol. 4, No. 10, p. 5.

Semich, J. William, “Program once, run anywhere. (Seer Technologies Inc.'s High Productivity Systems 5.0 computer-aided software engineering tool)”, *Datamation*, May 15, 1992, Vol. 38, No. 11, p. 87.

Kador, John, “Big GLs don't cry: overcoming a troubled past, 4GLs find acceptance on the RS/6000.” , *MIDRANGE Systems*, April 7, 1992, Vol. 5, No. 7, p. S21.

McLarnon, Scott, “4GLs getting new spin”, *Software Magazine*, March 15, 1992, Vol. 12, No. 4, p. 11.

Livingston, Dennis, “How integrators choose 4GLs.”, *Systems Integration*, July 1991, Vol. 24, No. 7, p. 49.

Cummings, Steve, "Gearing up for the object-oriented express.", *Corporate Computing*, June-July 1992, Vol. 1, No. 1, p. 285.

Coffee, Peter , "Lower life-cycle and maintenance costs make strong case for OOP.", *PC Week*, May 18, 1992, Vol. 9, No. 20, p. 133.

Moser, Karen D., "Complexity, costs slow acceptance of OOP approach: proponents tout increases in programmers' productivity.", *PC Week*, April 27, 1992, Vol. 9, No. 17, p. 61.

Guglielmo, Connie, "Object-oriented programming likely new norm for in-house development: programmers find OOP lets them produce better applications faster.", *MacWEEK*, March 23, 1992, Vol. 6, No. 12, p. 33.

Constantine, Larry, "Rewards and reuse.", *Computer Language*, July 1992, Vol. 9, No. 7, p. 104.

Marks, Donald, "Neutral observations. (Open Software Foundation's Architecture-Neutral Distribution Format)", *HP Professional*, April 1992, Vol. 6, No. 4, p. 70.

"The profit opportunities of smartsizing: beyond downsizing, beyond rightsizing.", *Datamation*, May 1, 1992, Vol. 38, No. 10, p. S2.

Keyes, Jessica, "New metrics needed for new generation: lines of code, function points won't do at the dawn of the graphical, object era.", *Software Magazine*, May 1992, Vol. 12, No. 6, p. 42.

CHAPTER 3. NEURAL AND FUZZY TECHNOLOGY TRENDS

SUMMARY

Future NASA spacecraft and robotic vehicles will be required to be increasingly autonomous, not only to enable them to operate in remote regions with minimal communications, but also to reduce the amount of human control so as to reduce operations cost. New methods are sought to advance the state of the art in autonomous control systems. Some relatively recent technology innovations are neural networks, fuzzy logic, and hybrid fuzzy-neural systems. These are sometimes called "soft computing" techniques. This report evaluates the uses of these methods as applied to autonomous systems, and discusses how they are complementary. Pattern recognition is found to be the key problem which links these new technologies.

The artificial neural network (or neural network) is a highly interconnected network of simple processing elements that has been found to be successful in artificial intelligence applications, especially pattern recognition. In pattern recognition, neural networks act as iterative search algorithms for finding a decision boundary in multi-dimensional feature space that separates data into clusters or classes. Neural network technology cannot be fully exploited unless the algorithms are implemented in large scale hardware networks. As parallel distributed computing algorithms, neural networks are suited to massively parallel architectures. The computing elements are extremely simple, or "fine-grained", and can thus be manufactured on a small scale. Neural network architectures are very scalable. It is possible to extrapolate the architecture to the molecular and even atomic scale.

Fuzzy logic is a method of applying a rule-based inference system (or "expert system") to quantitative data. Fuzzy logic has been successfully applied to complex control systems for which analytic solutions do not exist. Neural networks can be combined with fuzzy logic for hybrid systems to solve problems involving imprecision and uncertainty, such as are central to building autonomous systems.

1. INTRODUCTION

This report examines neural networks and fuzzy-neural systems as techniques for building autonomous systems. Neural networks will be examined primarily as a technique supporting pattern recognition for machine autonomy. Fuzzy logic will be examined as a technology for automatic control of complex systems, and as a pattern recognition technique.

Neural networks can be informally defined, following Kosko [1], as input-to-output "black-box" approximators. That is, given a training set of input sample data, paired with desired output responses, the network adapts its internal parameters to give an approximate map of the inputs to the outputs. After training, when presented with new data, the network interpolates to give a response consistent with its training. The ability of the network to "learn" is a key advantage for autonomous systems applications, since it implies an ability to adapt to a changing environment. Autonomous systems must recognize patterns in their sensor data in real time. Artificial learning is also important for handling complex environments, since these environments cannot be described in symbolic or deterministic form. Machines that learn can extract significance from complex sensor data without human programmers having to anticipate and code for all possible situations in advance.

Neural networks can be used as estimators for least-squares-error data classification, principal component analysis for feature extraction, data compression, and system identification for time-series prediction. The majority of applications of neural networks fall under the category of classification. The black-box function implemented by the neural network may be interpreted as a decision boundary or discriminant function in a multi-dimensional feature space. A feature is any measure used to characterize the signal. Each feature is assigned a dimension in the feature space.

The decision boundary partitions the space into regions in which the signal vectors cluster, each region representing a different class to which the member vectors belong. Neural network classifiers are mathematically related to statistical pattern recognition techniques, such as the Bayes optimal classifier.

When implemented in hardware, neural networks are parallel computers. They could be looked at as the kind of architecture that results from extending the "granularity" of processing to a limit. That is, the processing elements are reduced to the simplest possible function, and computation is distributed over a large number of these simple processors. In this architecture the connections assume greater importance. The connections themselves are processing elements. Signals passing along the connections are attenuated or amplified by differing amounts depending on an adjustable parameter called the weight of the connection. The weights can be thought of as storage elements in that the information that defines the estimated function resides in the weights. The simple nature of neural network processing elements makes them one of the few architectures capable of being scaled to an extreme of miniaturization, since for a given manufacturing technology and a given feature size, the simpler the processing element, the smaller it can be manufactured.

Fuzzy logic is an extension of set theory that allows sets with degrees of membership, or in other words sets with "fuzzy" rather than "crisp" boundaries. The degree of membership is described by a continuous function called the membership function. For example, in traditional set theory, a "crisp" set, labeled "fast" might be defined to include all automobiles traveling faster than 55 miles per hour. In fuzzy logic, a car traveling at 70 miles per hour might be defined to have 80 percent membership in the category of "fast", and 20 percent membership in the category "slow". Fuzzy logic has been successfully applied to complex control problems. It allows the designer to avoid seeking an analytic solution which may or may not exist, and instead to apply linguistic rules to continuously-valued sensor data. A fuzzy controller is essentially an expert system, a set of if-then rules, with the capability of being able to handle multiple simultaneous (ambiguous) rule-firings. The input sensor data activate multiple rules simultaneously, but the action of each is weighted according to a continuous-valued membership function, and the multiple results are combined into a single interpolated result. Fuzzy methods have led to new uses for neural networks in control systems. Neural networks can be used to extract the rules used by the fuzzy controller from the raw data, rather than requiring a human expert to provide the rules.

2. LONG-RANGE TECHNOLOGY TRENDS

2.1 ANALYSIS OF TRENDS

A mathematical model of a biological neuron was first proposed by McCulloch and Pitts in 1943 and was shown to have universal computational properties. The first neural network to become widely recognized was F. Rosenblatt's single-layer perceptron, invented around 1956. Rosenblatt also explored multilayer perceptrons with back propagation of errors in the 1950s. Hardware neural networks were built in the early years, such as for example that built by Joseph, et al. [2] in 1962. The number of research papers in the neural network field exploded in the early 1980s, prompted partly by a lack of success in the more mainstream fields of artificial intelligence, and partly by new successes in demonstrating the power of neural networks. In the 1990s, the field has continued to grow. While most research focuses on algorithms and software implementations, there is a significant amount of work on hardware implementations, mostly involving VLSI prototypes with multiple neural processors on a single chip. Other approaches include analog signal processing methods and photonic computing. The hardware implementations follow the trendlines of solid state technology in general.

In a hardware implementation, a neural network is a massively parallel processor in which the processors are massively interconnected. The network is built up from several simple building blocks: the processing elements, often called neurons or nodes, the connections, and the weights, often called synapses. Figure 3-1 shows an example of the forward signal flow through connections at one network node. (A separate set of connections would handle the feedback signal flow.) The node or neuron performs a sum-of-products function, followed by a thresholding or "activation" function. The nodes may be interconnected according to a variety of network topologies, with global or local connections, with single or multiple layers, and with or without feedback. Some of the most important network types are discussed in greater detail in the Appendix following this section.

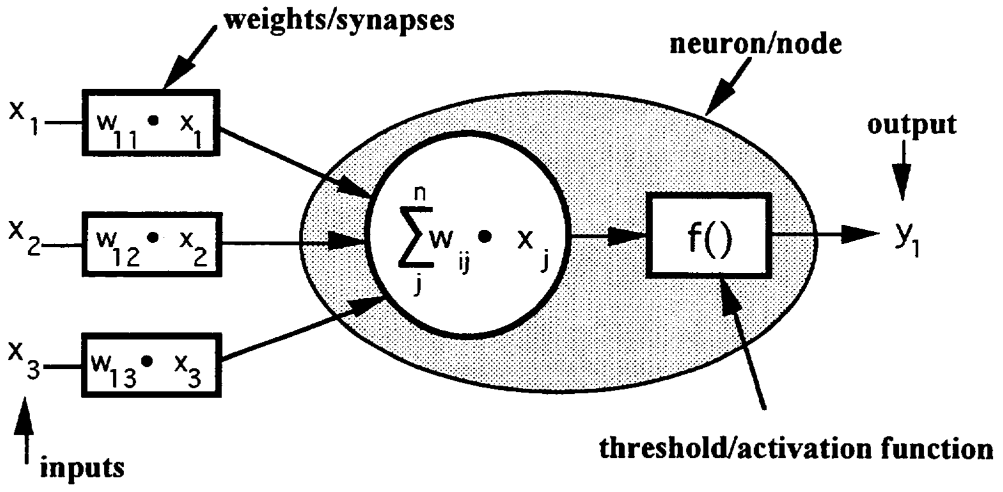

Figure 3-1. Neural Network Processing Element

Neural networks typically have two modes of operation: the training mode and the retrieval mode. In training mode the weights are made to relax to a state that minimizes the estimation error of the network through an adaptive search process called learning. Each network type has at least one distinct learning rule. To implement the learning rule, the network may be configured in a connection topology that is different from the retrieval mode topology. For example, for the multilayer perceptron using the back propagation algorithm, in training mode, estimation errors are fed back to the previous layer, where they are used to calculate new values of the weights. During retrieval mode the feedback paths are not used.

This architecture mimics the neurological structure of the brains of living organisms. The ultimate goal of neural network research is to be able to imitate some of the capabilities of living creatures such as recognizing recurring stimuli and understanding the sensory environment to a level sufficient to survive in that environment. The exploration of neural network architectures has led to some understanding of how the brain functions, but it is by no means certain that this approach will yield a full understanding. Aside from fundamental theoretical issues, there is the problem of sheer complexity. Present day technology does not permit the construction of neural networks even approaching the level of complexity of the human brain. A comparison of electronics technology with the human brain is shown in Table 3-1 [3].

Even though living organic neural synapses are a million times slower than silicon electronic devices, the human brain is estimated to perform 10^{16} operations per second compared to a supercomputer's 10^9 operations per second. At the same time the brain only consumes 10^{-15} joules per operation compared to an electronic device's 10^{-7} joules per operation. Carver Mead [4] is attempting to find explanations for the brain's immensely dense structure. His investigations

Table 3-1. Analog VLSI vs. Digital VLSI Computing Performance Comparison

	Power (watt/MCS)	Computation density (MCS*/cu.ft.)
Conventional digital	10	0.2
Special-purpose digital	0.1	10
Dedicated digital	0.002	40
Dedicated analog	$2x10^{-5}$	4000
Human brain	10^{-10}	10^{11}

* MCS = Million connection updates per second.

have led him to explore analog VLSI implementations of neural networks as a means of increasing device density. He hypothesizes that part of the explanation for the brain's performance is analog computation. Mead's artificial retina is an analog hardware neural network computer that has a 10,000-to-1 power consumption advantage over an equivalent digital computer. The implication is that analog computing allows a much higher device density, since power dissipation is a limiting factor in the ultimately achievable density. However, some experts caution that the power advantage of analog representations is highly dependent on the type of operation that is being performed [5].

While it is not possible to approach the complexity of the human brain with today's technology, there are some critical applications that are within reach of a full hardware implementation. Current development work is focusing on building massively parallel neural processors for real-time image pattern recognition that can be used for smart imaging sensors to be used on robotic vehicles such as missiles, unmanned aerial vehicles, or robotic surface rovers. The most common approach is to build a neural computer, based on custom VLSI chips that contain multiple processors, where each processor performs the calculations for one network node or neuron. A variety of prototypes have been described in the literature in recent years. A sample of these efforts is summarized in Table 3-2, below. There is not necessarily a trend in these data, other than the improvement in the underlying chip technology. It is difficult to directly compare the various chip designs directly because of the wide variations in architectures used. For example, some designers choose very simple integer-processing neurons, whereas others have implemented more general floating-point processors.

VLSI implementations are not very scalable to higher numbers of neurons. It is difficult to increase the number of neurons and connections beyond a certain point because chips are limited in the number of on-chip and chip-to-chip connections. This is due to the fundamental limits of a 2-dimensional architecture. To scale beyond a certain point, one of two alternative strategies must be adopted. One strategy is to pursue development of locally connected architectures, or cellular networks, in which neurons are connected only to local neighbors. In this class of network, of which Mead's neural retina is an example, the number of connections scales linearly with the number of neurons. For more general networks, however, the number of connections scales as the square of the number of neurons, so that a 3-dimensional architecture must be developed. Three dimensional architectures usually require some form of photonic computing, because photonic methods allow 3-dimensional network interconnection. In photonic architectures, parallel signals can be launched from 2-dimensional arrays and propagated through a free-space third dimension.

Table 3-2. VLSI neural network Chips

Developer	Complexity	Technology	Performance
C. Mead, et al. (Cal Tech)	8K neurons	3 micron analog CMOS	50 frames/sec.
A. Chaing, et al. (MIT Lincoln)	500 neurons, 64K weights	analog CCD	5x10^9 cps*
M. Holler, et al. (Intel)	64 neurons, 8K weights	1 micron CMOS	10^{10} cps
Mitsubishi	336 neurons, 28K weights		10^{10} cps
H.P. Graf, et al. (Bell labs)	54 neurons, 3K weights	2.5 micron CMOS	10 MHz clock
G. Works, et al. (SAIC)	8 neurons, 64 weights	1 micron CMOS	40 MHz clock
Hughes	32 neurons, 1K weights		100K patterns/sec.
Adaptive Solutions	64 neurons		10^{10} multiply-accumulate operations per second

* Connection updates per second

Neural networks appear to be a candidate architecture for scaling massively parallel computers to the million-processor level. To build million-processor computers requires hundreds to thousands of processing elements per chip and thousands of chips. To build even hundreds of processors per chip requires extremely simple processors. Neural networks use very simple processing elements, and thus may be suitable. However, neural networks require extremely dense interconnectivity, leading to the need for solutions such as optical interconnection. Figure 3-2 shows the possible evolution of multiprocessor chips, and compares the density over time of several different technologies. In the near term, there are digital chips with multiple neural processors.

An early photonic implementation of a neural network (a Hopfield type) was an analog vector-matrix multiplier using incoherent light [6]. The input layer consisted of a linear array of light-emitting diodes (LEDs), and the output was a linear array of photodetectors. The light from each LED was spread by means of anamorphic optics to fall on one column of a 2-dimensional array binary mask. Those pixels of the mask representing a weight of +1 were transparent, so that light passing through represented a multiplication by a weight of +1. Those pixels representing a weight of 0 were opaque, so that the lack of light in this optical path represented multiplication by 0. The final summation of all the weighted products was accomplished by focusing the light through another anamorphic lens system to the linear array of photodetectors. Each photodetector received the weighted sum from one row of the weight mask.

This architecture can in principle be extended to moderate network sizes (approximately 1000 neurons), but is ultimately limited because of the 1-dimensional nature of the input array. In order to scale to larger networks, it is necessary to implement 2-dimensional input and output arrays. To do this, instead of one 2-dimensional optical mask, a set of 2-dimensional optical masks is needed, one for each input row. This implies that some form of multiplexing is needed. The commonly accepted solution is angular multiplexing. This can be accomplished by means of a volumetric phase hologram operating on coherent light. A volumetric phase hologram can theoretically

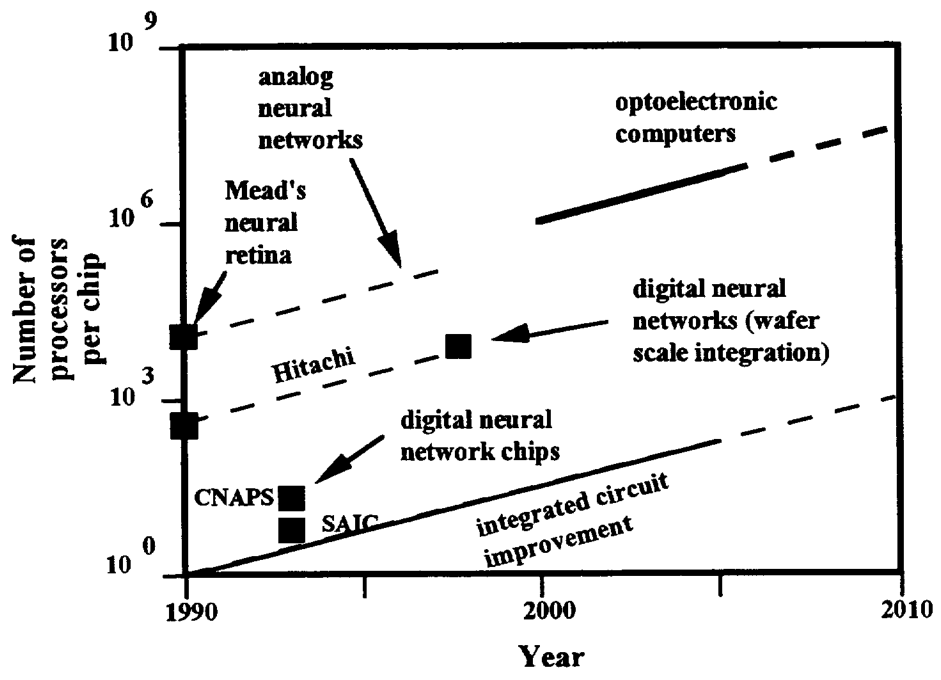

Figure 3-2. Evolution of Multi-processor Integrated Circuits

achieve an arbitrary mapping from a set of coherent plane wavefronts impinging on the hologram from different angles to any desired set of plane wavefronts at another set of angles emanating from the other side of the hologram. Unfortunately, the practical implementation of such volumetric holograms is still in the fundamental materials research phase. The materials characteristics required for this function are identical to those required for successful volumetric holographic storage. The most promising materials for both applications appear to be photorefractive crystals. The long-term target for photonic and optoelectronic implementation complexity is 10^5 to 10^6 neurons with 10^{10} interconnections [7].

2.2 FUTURE APPLICATIONS

NASA's OSC is interested in reducing costs by using AI to automate operations. For example, the more autonomous a spacecraft is, the more mission control operations could be simplified. Likewise, application of AI to mission control itself could reduce the staff required to support missions. In this section we will focus on applications suited to autonomous robotic spacecraft and ground vehicles, including both on-board and ground-based control portions. Methods that can be applied to building autonomous systems include:

- Pattern recognition and feature extraction
- Automated fault diagnosis
- Image analysis
- Advanced memories
- Fast optimization
- Advanced controllers.

2.2.1 Pattern Recognition and Feature Extraction

There are many applications of pattern recognition that can be applied to automating NASA's OSC systems. Pattern recognition, especially as applied to sensor data and imagery, is fundamental to the implementation of autonomous vehicles and autonomous control systems. A primary function of pattern recognition for space systems is the classification of sensor data into meaningful sets that enable an autonomous controller to make decisions based on its environment and the state of its own resources. Automatic recognition such as this permits the construction of both intelligent controllers for spacecraft and automated expert assistants for ground control. Many recently designed spacecraft incorporate this kind of automation to some degree.

Neural networks are primarily pattern recognizers. In particular, the multilayer perceptron has been found to be a highly practical and universal parallel computing architecture for implementing many proven statistical pattern recognition techniques. The paradigm for statistical classifiers is the Bayes classifier algorithm. The back propagation multilayer perceptron neural network can calculate the least mean squares approximation to the Bayes discriminant function, and thus can be used as a Bayes classifier. One study showed that the two methods generally have comparable results, but that back propagation may be preferred when the statistical properties of the input data are unknown [8]. Another study compared a back propagation multilayer perceptron to two standard non-Bayes classification techniques, a minimum-mean-distance classifier and a nearest-neighbor classifier. The multilayer perceptron outperformed the minimum-mean-distance classifier and performed as well as the nearest-neighbor classifier under noiseless conditions, but outperformed both of the other classifiers under noisy conditions. It was also found that the computational complexity for the neural network was 100 times less than for the nearest-neighbor classifier [9].

An important part of pattern recognition is feature extraction, the problem of finding a representation of a given set of data which best captures the signature or discriminatory features of the data, while eliminating uninformative data. This process is a prerequisite for solving the classification problem. Feature extraction is usually done "by hand" using intuition and prior knowledge of the data. While there are no general rules for feature extraction, certain techniques have been developed over the years to automate the process. Multilayer perceptrons have been applied to these techniques. In particular, a perceptron can implement principal component analysis, an important tool in data analysis for feature extraction. Application of this method allows one to find the most important features of the data by rotation of the coordinate system. In the perceptron implementation, the input and output training vectors of the network are identical. The hidden layer of the network has fewer nodes than either the input or output layers. The network is therefore constrained to find a data-compressed representation of the information contained in the input vector [10].

2.2.2 Automatic Fault Diagnosis

An example of how neural networks might perform pattern recognition for automating future spacecraft operations is a combined neural network and expert system satellite downlink fault analyzer developed by Science Applications International Corporation. In the downlink that was studied, different sources of external interference or internal faults generated different bit error patterns. The neural network was trained to recognize or classify the error patterns, and the classification decisions of the neural network were input to a diagnostic expert system for a final decision on the cause of the problem. This architecture appears to be suited to a wide variety of applications for automating operations. The neural network part of the system reduces the data, and converts complex data sets to symbols standing for classes of phenomena. These symbols can then be manipulated by expert systems, fuzzy expert systems, decision trees, case-based reasoning, or other symbolic AI techniques to derive inferences about the patterns, and to arrive at decisions. The classifications made by neural networks may be inherently fuzzy. The output of a neural network can therefore be a natural match to a fuzzy logic system. In these approaches, the neural network is a means of finding hidden patterns in complex data.

Another example of automated fault diagnosis is diagnosis of mechanical systems by means of vibration signature. Neural networks have been used successfully for acoustic signature recognition. The low sampling rate for acoustic data makes it relatively easier to build real-time neural network analyzers for this application. When faults can be predicted by means of a telltale acoustic signature, a neural network can be trained to "recognize" such a signature. The signature can be extracted through the training process even if not readily observable to the human observer.

2.2.3 Image Processing

Image classification is an important category for neural network applications since the large dimensionality of images demands massively parallel computing such as can be provided by neural network architectures, in order to achieve real-time results. Some recent examples of neural network methods applied to satellite imagery analysis are described below.

A paper presented at a recent NASA conference [11] reports on a probabilistic neural network for image segmentation and compares the performance to a Gaussian maximum-likelihood classifier and a back propagation multilayer perceptron. The probabilistic neural network has much faster training characteristics than the other two approaches, along with higher accuracy than the maximum-likelihood classifier, but lower accuracy than the multilayer perceptron. It has inferior classification speed and high memory requirements, since it must store the entire exemplar set. These characteristics have led the authors to propose it for use in dynamic imaging.

Another paper [12] reports on a minimum misclassification error neural network and compares it to the standard back propagation approach. The authors apply it to multispectral earth imagery pixel classification, and find a significant improvement in error rate using their approach.

A third paper [13] is representative of the trend toward the use of hybrid systems. The system used two neural networks and merged the results of the two with a third type of network. One network is a standard back propagation multilayer perceptron. The second network is a neural network implementation of a binary decision tree. Binary decision trees are a standard hierarchical pattern classification technique, but the use of neural networks to implement them is a recent development. This system was applied to processing of data from a multiple spectral band passive microwave radiometer flown aboard the Defense Meteorological Satellites. The performance from combined independent classifiers gave 94 percent accuracy compared to 80 percent and 88 percent, respectively, for the two networks operating separately.

Attractor networks similar to the Hopfield network and the Boltzmann machine (see Appendix) have been applied to problems such as "early vision", that part of image pattern recognition in which a possible object is extracted from background and foreground. An attractor network algorithm can be used to find a fast solution to a maximum a posteriori conditional probability estimate of the object's surface. Successes with this technique have been considered to be a breakthrough for applications such as automatic target recognition [14].

A related problem is texture segmentation. Texture segmentation is the automatic partitioning of an image into contiguous texture regions. Texture segmentation is a major research topic in the field of pattern recognition and image processing, and has many applications such as for image compression and object extraction. Texture segmentation is of particular interest to NASA, since it is a popular and well-established means of automatically classifying multispectral remote sensing imagery according to image content [10]. The diversity of neural network architectures has proven to be fertile ground for exploring fine-grained parallel models of the human visual system. Neural network models of texture segmentation are useful for validating early vision hypotheses, since it is found that such simulations can recreate characteristics of the human early vision processing. These results are important for progress in robotic vision systems [8].

2.2.4 Advanced Memories for Robotics

The interest in the Hopfield neural network as an associative memory (see Appendix) was spurred mainly by the fact that memories which operate by associative recall are needed to support artificial intelligence and robotics, since these applications require information to be recalled based on the cues derived from the status of the artificial agent, rather than by item-by-item search of a database. For example, robotic rovers will benefit from a kind of memory which recalls information based on similarity to the rover's current surroundings as measured by its various sensors. This kind of associative recall is useful for databases containing non-symbolic information such as images, since the information can be recalled based on content rather than index information which cannot adequately describe the content.

Networks of the Hopfield type seem to be a preferred architecture for those exploring alternatives for very high density storage because of the analogy with spin glasses. Spin glasses can be thought of as storing information at the atomic scale in the form of the spin states of the atoms. Use of the mathematics of spin glasses to solve computational problems was first done by Scott Kirkpatrick, an originator of one of the spin glass models, and his colleagues at IBM. Their method is called simulated annealing. At about the same time, John Hopfield invented his network using similar mathematics to show that analogs of spin glass systems could perform both storage and computation [15].

While use of the Hopfield network as a practical content-addressable memory remains a laboratory exercise, there is a substantial body of opinion which holds that this is a promising model for computation at the molecular and atomic scale. The reasoning is that since the Hopfield network shows that disordered arrays of simple on-off elements can perform useful computational functions, and since spin glasses are also arrays of units with binary spin states, it may someday be possible to use spin glasses as atomic-scale computers and storage devices. At the forefront of research in this field is the work being done on spin glass materials, with an aim toward using these materials for storage and computation. A recent article discusses the work of P. Chandra at NEC Corp. whose goal is storage at the atomic scale, with the spin states of individual atoms representing bits of data [16].

2.2.5 Fast Optimization

Planning and scheduling of resources are fundamental activities of autonomous systems. Schedules are detailed implementations of plans. An intelligent controller would be able to derive a schedule from a plan. A more advanced controller would be able to create plans given a set of goals. Neural networks can play a role as a component in planning and scheduling. An important part of many planning and scheduling problems is the optimum allocation of resources. The traveling salesman problem is a mathematical generalization of a class of resource assignment optimization. The problem may be briefly stated as finding the shortest route that connects a set of N sites to be visited. This involves minimizing an energy or cost function associated with traveling the route, where the number of possible routes grows combinatorially with N. The traveling salesman problem arises in many aerospace applications such as assigning multiple targets to multiple interceptors, calculating tracks for multiple radar targets [17], and for multi-sensor fusion, which is the problem of correlating tracks from multiple sensors [18]. The Hopfield neural network has been shown to be able to find good approximate solutions to the traveling salesman problem very quickly.

Kohonen feature map networks (see Appendix) can also be applied to the traveling salesman problem. In this approach, the Kohonen network maps a 2-dimensional coordinate plane containing the coordinates of the sites to be visited to a 1-dimensional ring, which represents the tour to be taken. Training makes the weights converge toward values equal to the coordinates of the sites, while the topology-preserving property assures that the distance between adjacent points on the ring represents input coordinates that are also close together. Pictorially, the behavior of the network shows the ring behaving like a rubber band that gradually stretches to touch all the sites to be visited while keeping the amount of stretching to a minimum. For this reason it has been called an elastic ring method [19].

2.2.6 Advanced Controllers

Fuzzy logic has been shown to be an important advance in the field of control systems. The success in applying fuzzy logic to consumer electronic devices, such as video cameras, ovens, washing machines, and air conditioners, has been well-publicized. Less well-publicized are successes in vehicle control problems such as for automatic trains, the docking of ships, and helicopter control. Fuzzy logic is useful for designing control systems for complex systems which are difficult or expensive to model by means of more traditional mathematics. It allows the application of fuzzy categories that are intuitive, such as fast/slow, hot/cold, small/large, along with intuitive human concepts of control, providing a "user-friendly" analysis tool for complex control problems.

Fuzzy controllers have been shown to give better control performance than traditional controllers for robotic manipulators. The differential equations for a robotic arm are nonlinear, and require a number of specialized techniques, each one addressing a different part of the solution, but none totally complete. Fuzzy logic allows for the design of a simple controller, without the need for a complete mathematical solution. In a recent demonstration [20] a fuzzy controller outperformed a proportional controller. It was able to move the robotic arm between two points smoothly without overshoot. The proportional controller exhibited overshoot and was unable to correct for the smallest incremental errors, whereas the fuzzy controller was able to move by the amount of a single minimum increment.

Perhaps the most sophisticated fuzzy control system built to date is the work of Dr. Michio Sugeno of the Tokyo Institute of Technology. This is a controller for a model helicopter using a fuzzy expert system consisting of 384 rules derived from the mathematics of helicopter motion and the expertise of helicopter pilots. The difficulty of this problem and the success of fuzzy logic in solving it validates the power of the technique [21].

2.3 EMERGING TECHNOLOGIES

Neural networks can be combined with fuzzy logic to improve the ability of the system to handle objects or classes with fuzzy boundaries, which includes most real-world AI problems.

2.3.1 Fuzzy Pattern Recognition

A fuzzy approach to the weighing of evidence was applied to pattern recognition [22]. The researcher constructed a fuzzy rule-base for logically weighing and combining elements of evidence. This rule-base is a general procedure for quantitative evaluation of evidence. The procedure was then applied to alphanumeric character recognition. The elements of evidence for characters were the presence or absence of features such as line segments, arcs, angles, and connection locations. Each character exemplar was defined by a template consisting of a set of such features. Fuzzy quantization was used, such as long segments, slightly sloping segments, or a connection located roughly in the middle. These fuzzy labels were applied to overlapping ranges of quantitative measures, which is typical of fuzzy applications.

2.3.2 A Fuzzy Neural Network

Simplified Fuzzy ARTMAP (SFAM) [23] is a recently developed neural network that trains very quickly compared to back propagation neural networks, and, unlike the latter, has only one user-defined parameter. It is therefore much easier to use than a back propagation neural network, while being an equally capable general pattern classifier. It makes use of fuzzy logic to simplify its internal mathematical operations. The name of this network derives from its history. The term adaptive resonance theory (ART) was the name given to a class of unsupervised learning networks developed by Carpenter and Grossberg to which they later applied fuzzy techniques, hence the term "fuzzy ARTMAP". Recently, the architecture was further simplified, resulting in the SFAM.

As SFAM is trained, it automatically decides whether to classify a new pattern as a member of an existing class, or to create a new class, and add new weights and connections to accommodate the new class. It therefore adjusts its own size to accommodate the data, so that no guesswork is required by the user as to what the proper size should be. Fuzzy logic is used in deciding how close a match exists between the training data pattern and its exemplar. Fuzzy logic is also used in the weight update rule. Specifically, a fuzzy AND is used, which consists of simply selecting the minimum of the two operands. This simple operation is claimed to result in performance equal to that of more complex learning rules. In this case fuzzy logic is not used with a fuzzy rule base as in most applications, but rather is used to enhance the performance of an existing neural network architecture.

2.3.3 A Fuzzy Neuro-Controller

For well-understood control problems, the fuzzy logic method may be easy to apply, but for more complex situations, the method runs into two practical difficulties. First, it is necessary to implement a control strategy in terms of specific rules. This may be difficult to do for a system whose behavior is not fully known, or for which there is no human expertise, or is described by a large quantity of data that is difficult to comprehend intuitively. Second, it is usually necessary to fine-tune the membership functions based on trial and error. It may be very difficult to determine how to do this fine-tuning from the analysis of the raw data describing the fuzzy controller's performance. A recent article describes how neural networks may be applied to these problems in a combined neural network and fuzzy controller, or "fuzzy neuro-controller" [24]. The result is a network that not only solves these problems, but is capable of adjusting from experience.

The system described by the author consists of two steps, each performed by a different neural network. The first step is to find the rule base for the fuzzy controller by analysis of input and output data pairs. This is done by means of a two-stage neural network. The first stage is a Kohonen self-organizing map, which finds centroids of clusters of the data. These clusters represent the fuzzy sets. Since a Kohonen net can only achieve sub-optimal clustering, a second stage consisting of a Kohonen learning vector quantization network, improves the cluster definition. The resulting clusters are used to derive rules relating outputs to inputs.

In the second step, a fuzzy controller is implemented as a six-layer back propagation neural network. Each step in the fuzzy logic method can be translated to an equivalent neural network operation. The neural network architecture allows the system to be trained using back propagation learning. This allows the network to fine-tune its fuzzy membership functions over time, as it is exposed to additional real-world data.

3. RESULTS

3.1 IMPACT OF NEURAL AND FUZZY TECHNOLOGY ON NASA

In general, AI will be an enabling technology for NASA. It will allow things to be done that could not be done otherwise. Applied to real-time control problems for spacecraft and robotic vehicles, AI will make these vehicles more autonomous. Vehicle autonomy will allow robotic exploration of environments where it is too dangerous or too expensive to send human beings. AI applied to complex control problems will allow new kinds of aerospace vehicles to be built such as single-stage-to-orbit reusable vehicles.

Neural networks can be considered to be a category of AI. Neural networks provide powerful tools for analyzing complex data sets derived from sensors operating in the real world without resorting to models of the world. In this sense they differ from other AI methods which often start out from the assumption of a model.

While there is an enormous amount of research in the field of neural networks, most of it still has an academic orientation. Many of the neural networks that have been applied commercially are small networks. Neural networks are expected to come into their own in applications where large, real-time networks are needed, but the hardware technology is not yet available for this.

Since neural networks are general scalable parallel algorithms, they will contribute to the expansion of applications for parallel computers. The wide variety of applications to which neural networks have been successfully applied is an indication that it is worthwhile to pursue the scaling of parallelism to millions of processing units.

Neural networks may have a large impact on one of NASA's biggest problems, which is the large amount of image data that will be generated from low earth orbiting remote sensing satellites. Neural networks are being proposed as a means for automated classification of terrain regions within images. Neural networks will therefore probably have a major influence on the design of future parallel image processing computers, and on the design of software for these computers.

As stated earlier, fuzzy logic technology is primarily applied to designing control systems. Since most fuzzy control algorithms are simpler than their equivalent traditional controllers, they can be easily implemented in single-chip microcontrollers. The success and increasing popularity of this method are indicative of a trend, and it seems certain that fuzzy controllers will appear in many future NASA projects. At least some aspects of fuzzy control will begin to be used in control systems for spacecraft, antenna gimbals, and robots, for example. Of interest to NASA is a Japanese project to design a fuzzy controller for helicopters [21]. This project indicates that fuzzy controllers could be applied to many aircraft and spacecraft control problems.

We have also seen that principles of fuzzy logic are being incorporated into neural networks. This effort is producing simplifications to the existing algorithms, which has a corresponding impact on processing speed. Also, neural networks can sort data into overlapping clusters that are essentially fuzzy sets, and thus can be used as a pre-processing stage for fuzzy processing. This approach may allow fuzzy methods to be applied to more complex systems for which a rule set does not exist.

Fuzzy logic appears to offer real benefits for certain kinds of problems. There is an indication that it can be successfully applied to difficult nonlinear control problems that have not fully yielded to traditional methods. There are also indications that the concept of fuzzy set membership allows one to process a large class of real-world domains which have an inherent uncertainty or "fuzziness". Human intelligence does not seem to use rigorous quantization of the signals it receives. In character recognition quasi-topological features have been shown to be successful, that is, features that have both a fuzzy metric aspect and a topological aspect. Features may vary in size, orientation, or be slightly distorted. Features may vary slightly in location. The concept of fuzzy sets seems to capture at least one aspect of this variability. It will therefore become an essential tool in the further development of "machine intelligence" systems. It may also serve to inspire new approaches to handling the variability encountered in AI problems.

3.2 TECHNOLOGY ROADMAP

The following is a preliminary technology roadmap, or possible timeline of major developments in the field of neural networks and fuzzy-neural systems over the next 25 years.

1996-2000: During this period, neural networks will increasingly be integrated into complex sensor-based and autonomous systems. The next few years will see an increase in the use of hybrid systems, that is, combinations of neural networks and expert systems, fuzzy logic, genetic search algorithms, decision trees, and others. Neural networks will be used as a front-end processor to sort noisy data into classes for processing by a different form of AI. Another form of hybrid will analyze data by parallel independent algorithms and combine the independent decisions to achieve higher performance (data fusion).

An important advance for neural networks will be the migration to low-cost, real-time commercial applications. Inexpensive, high performance (hundreds of MIPS) processors will be an enabling technology for this advance. Improvements in user-friendliness will be required to help determine which networks to apply in what circumstances, how to relate different data clustering configurations to network size and type, how to determine the hidden layer size, and how to verify performance given a limited training set.

Fuzzy logic will become more widely recognized and more widely understood as a powerful tool for AI applications. It will be in routine use by engineers for the design of control systems, for improving the performance of knowledge-based systems, and for "fuzzifying" data sets in pattern classification problems. New applications will proliferate. Neural networks may improve the ease of use of fuzzy methods. More powerful and general fuzzy methods may be discovered.

2000-2010: Massively parallel computers will be used on-board spacecraft. Early-vision massively parallel processors will be built in to image sensors for remote sensing and robotic exploration. More advanced neural network-based autonomous systems that can learn to recognize objects and navigate through a complex environment will be developed. Such networks will need the capability to "grow" and "prune" themselves in a dynamic environment, and to learn "on the fly". Networks will take into account multiple sources of information for object recognition, including especially object segmentation based on cues from motion, perspective, and color.

In terms of implementation, hardware architectures will follow two main strategies: two dimensional architectures for locally-connected networks for image processing, and 3-dimensional architectures for both locally- and globally-connected architectures. Locally-connected architectures may be adequate for most early-vision processing problems. On-chip parallelism will increase to thousands of neurons per chip.

Fuzzy sets will be a standard data type in software. More powerful and general fuzzy methods may become widely used. The example of fuzzy logic will propel new methods for mathematically describing objects that have no exact quantitative description, a set that includes most objects dealt with by human intelligence.

2010-2020: New technologies may become available for implementing neural networks, including molecular and atomic scale devices. The validity of neural networks as a model of brain function may begin to be explored experimentally using massively parallel computing at the molecular scale. The neural network model will be an important example for the programming of this technology, since explicit, line-by-line instructions will not be practical for such complex systems. Photonic methods may come to maturity, and could be used to exploit 3-dimensional architectures.

3.3 FEASIBILITY AND RISK

The question arises as to merits of neural networks over other approaches. There is evidence that neural networks are being used in commercial applications where large profits can be made or lost depending on a few percentage points difference in performance, such as in the field of optical character recognition. This implies that those in a position to evaluate the commercial usefulness of the technology, the product developers themselves, believe that there is a true advantage to using neural networks.

The relative ease of use of neural networks may have to do with the fact that neural networks are a "brute force" technique that encourages the user to forego the detailed manual analysis of the data that is typical in statistical pattern recognition [25]. This can be risky, since it sometimes happens that systems that perform well on training data, perform poorly on new data when there is an unexpected and undesired feature on which the network has converged. In this sense, neural networks can exhibit brittleness, as do all other forms of AI. Neural networks do not relieve the designer from the need to be familiar with the properties of his data sets.

Although neural networks may suffer from brittleness in a certain sense, there is another sense in which neural network decisions are potentially more robust than expert systems. They have the ability to make correct decisions in the presence of noise. They also have the property that since computation is distributed among a large number of units, the loss of any one path or unit will not necessarily have a big impact on the final output. Neural networks also have the interesting advantage of being applicable to both the feature extraction problem and the classification problem, both essential parts of pattern recognition.

Much remains to be done in the theoretical arena. Given the profusion of neural network types, there are no clear guidelines as to which neural networks should be used for a given problem. There are many pitfalls in using neural networks. A serious problem is that back propagation networks scale poorly in training time as size increases. This is a manifestation of the fact that neural networks suffer from the "curse of dimensionality", the combinatorial growth of searches with respect to input vector dimension common to all pattern recognition problems. There are information theoretic questions such as determining how much training data is enough to represent the discriminant function, and how many hidden units are best. There is the problem of visualizing clusters in multi-dimensional space, and determining whether the network has converged on a valid decision boundary.

Pattern recognition with neural networks is as much an art as it is using other techniques. That is, most of the work is taken up in data-specific aspects such as feature selection, preprocessing, and manual analysis. There are aspects of the pattern recognition problem for which neural networks are no cure, since they cannot be addressed by any technique. For example, there is no "best" feature representation. It is theoretically impossible to inspect even a small fraction of the possible representations. It is also a general problem to decide how many features to use. Nor is there a "best" classification, since for a given unclassified set of points in a multi-dimensional space, there is no unique clustering solution. Different clustering algorithms give different clusters. The best practical approach is to apply several techniques and look for common clusters. Choices must be made primarily on the basis of intuition, trial and error.

As with any new technology, there is a risk in properly evaluating fuzzy logic's capability and the range of its proper application. Many grandiose claims have been made regarding fuzzy logic, but the current applications are fairly limited. The importance of fuzzy logic undoubtedly lies in its ability to handle data that does not fit into discrete categories.

Besides its strengths, fuzzy logic clearly also has weaknesses. One problem is that one has to have some set of rules, a "control strategy", prior to using fuzzy logic. This set of rules usually comes from pre-existing human expertise or intuitive knowledge. Such expertise is not always available. Furthermore, not all human capability can be described by a set of rules. This has been the source of many difficulties in the field of AI.

Another weakness is that some of the procedures in the fuzzy method seem arbitrary, such as the process of defuzzification. This does not prevent it from working well. However, it seems possible that there may be some more general mathematical method that would not have this arbitrariness.

A fuzzy controller often requires adjustment of parameters such as the membership function shapes. This requires a great deal of experimental data, either from simulations or actual prototypes. Once the designer has collected a large set of input-output data acquired from such experimentation, the question arises, why not use a neural network? In other words, if one knows enough about the system to simulate its inputs and outputs, is a rule-based system really appropriate? One could instead train a neural network on the simulation data. This question seems especially pertinent in light of the fact that researchers are mapping the fuzzy logic method to neural network architectures.

To summarize, the fuzzy logic method seems sufficiently mature to be of low risk when applied to control problems for which a control strategy is known. Fuzzy logic methods also offer a means of speeding up and simplifying the design process, while improving performance. For more complex problems for which a control strategy is not known, it appears that further research must be done. Outside the realm of fuzzy controllers, the concept of fuzzy sets has proven its usefulness, particularly as a way of reducing the brittleness of expert systems that operate on quantitative inputs. It seems certain that many future automated systems that operate on physical measurements will incorporate some aspect of fuzzy logic. Fuzzy logic offers a new way to deal with inexact and non-quantitative objects.

APPENDIX.

An Approximation Network: The Multilayer Perceptron

The multilayer perceptron was first proposed in the 1950s, but only became widely used in the 1980s when a practical learning algorithm called back propagation was widely disseminated. It can be classified as an approximation network, because it is used primarily to calculate an approximation to the function describing the training data input to output relationship. The topology of the multilayer perceptron is illustrated in Figure A-1, which shows a three-layer network having two hidden layers and one output layer. The input layer is usually not counted since it performs no function other than fan-out. Each node in the network performs a sum-of-products function, followed by a nonlinear activation (threshold) function:

$$y_i = f(\Sigma_j w_{ij} x_j + T),$$

where y_i = output of ith node

$f()$ = activation function

w_{ij} = weight in the i-jth connection

x_j = jth input

T = threshold

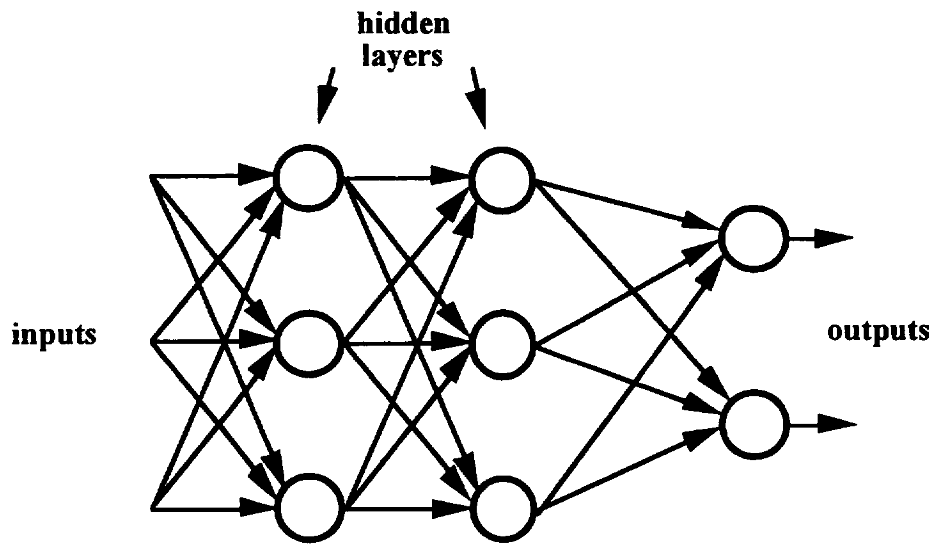

Figure A-1. A Three-Layer Neural Network

Taking all the i nodes together, forming a "layer", the operation is equivalent to a matrix-vector inner product, i.e.,

$$\mathbf{Y} = f(\mathbf{W} \bullet \mathbf{X})$$

where $\mathbf{Y}$ = output vector of the layer

$f()$ = activation function

$\mathbf{W}$ = weight matrix

$\mathbf{X}$ = input vector

The multilayer perceptron has two modes of operation, a feed-forward mode and a feedback mode. The feedback mode is the training mode, in which the weights are adjusted toward a convergent solution based on training data. The multilayer perceptron requires supervised training. That is, for each training iteration an input vector belonging to a pre-determined class is paired with the desired output that encodes the correct classification decision. When the input is presented to the network, the actual output is compared to the desired (training) output to generate an error measure. This error is fed back into the network to update the weights. The popularity of the network results from the fact that the back propagation algorithm is a practical method of translating errors at the output to meaningful error measures at the hidden layers, so that the weights in the hidden layers can be adjusted. The adjustment is in the direction of the negative gradient of the global error surface. Back propagation is therefore a type of gradient search algorithm.

The primary application of the multilayer perceptron is as a signal classifier with learning. Rosenblatt's original single-layer perceptron has the capability to implement a maximum likelihood Gaussian classifier, a basic statistical pattern recognition function which finds an optimum decision boundary between uncorrelated Gaussian-distributed clusters. However, it can only learn planar boundary functions. Multilayer perceptrons have more general capabilities. With the addition of one hidden layer, a simple curvilinear boundary can be learned by the network. With two hidden layers, an arbitrary boundary may be learned [26].

Besides signal classification, the multilayer perceptron is suitable for regression curve fitting, modeling nonlinear functions, feature extraction, and data compression. Successful applications include signal classification problems in speech recognition, character recognition, sonar, radar, seismography, electrocardiography, and automatic target recognition. It has also been applied to adaptive control problems such as robotic hand-eye coordination.

A Single-Layer Attractor Network: the Hopfield Network

The Hopfield network is the prototype of an important class of networks based on the Ising model of the dynamic behavior of a spin glass. A spin glass is a lattice of atoms in random spin states that converge to a stable configuration, known in dynamical systems theory as an attractor. In the Hopfield network there is only one layer of nodes, each representing the dynamical units of the Ising model. For a given initial condition and set of weights, the network will settle to a stable state after several feedback iterations. For a given network there are a number of stable states (attractors) of the network. These stable states can be thought of as storing information in an associative sense. For example, if a noisy version of one of the stored states (in the form of a binary vector) is input to the network, the network will converge to the nearest stable state, which is the stored vector it most resembles. This convergence results in an output which is the original noise-free version of the input. Similarly, a stored vector can be retrieved using only part of that vector. The network can also be looked at as a classifier that compares the input with the stored exemplars, and responds with the most closely matching exemplar.

The Hopfield network has been the subject of a great deal of interest as a content addressable memory. However it has drawbacks for this application. First, it is limited in storage capacity in relation to the number of nodes, but especially in relation to the number of connections and weights. The number of connections is the square of the number of nodes. In software simulations, this translates to a relatively high number of computations. Secondly, the stored patterns must be significantly different from one another. A related problem is the problem of spurious states. There are states of the network that do not correspond to any stored pattern, which result in erroneous responses. An attractor network that overcomes some of the above-mentioned problems is the Hamming network [27 - 31].

A trend which emerged in the 1980s was that the Hopfield net became a vehicle for many early projects to implement neural networks in hardware parallel architectures on VLSI chips. Possible reasons for this were the simplicity of the topology and the fact that it operates on binary data. It also is capable of being implemented as an analog computer. The computing elements become even simpler in this case. A school of thought emerged that favors analog implementations of neural networks for the purpose of achieving higher circuit density on a single chip network.

Other Important Networks

A statistical attractor network usually has a Hopfield topology, but follows stochastic rather than deterministic dynamics, typically using a Boltzmann probability distribution. As a result, a new parameter is introduced into the model, that of "temperature". The network dynamics then executes according to a statistical mechanics model. The network is able to perform an optimization computation by relaxing from a high energy state to a globally minimum energy state. A problem with this random search is that it often gets stuck in a local minimum. The problem is alleviated by means of a learning algorithm using simulated annealing, in which the temperature parameter is varied during training. By lowering the temperature gradually, the system almost always escapes the local minima and reaches the global minimum. This type of network is useful for fast, approximate solutions to difficult optimization problems. The current trend is toward the use of newer network types, such as those using the mean-field annealing enhancement [32].

The Kohonen network is called a self-organizing feature map since it finds a mapping between input and output spaces without supervised training. It typically maps an input vector of dimension n to a 2-dimensional array. During training, each input vector tends to activate only one output node in the 2-dimensional array. Which node gets activated depends on a competitive learning rule that involves lateral interaction between the output nodes. After convergence, the competition has settled and only one node is the winner for each class of the input vectors. This network has the valuable feature that it is sensitive to topological order. After training, nodes in the output layer that are spatially close together are sensitive to input patterns that are similar. The spatial location of the output nodes becomes a map of the features of the input vectors. The Kohonen network is equivalent to the earlier K-means clustering statistical pattern recognition algorithm [33].

This network is useful for data visualization, for extracting the implicit order in a set of statistically related data. Its principles have been adapted for use in a variety of applications. However the basic network is not an optimum classifier, and Kohonen recommends against using it for signal classification. He proposes modifications that make it more suitable for this application. For example, he developed a network called supervised competitive learning for speech recognition applications [34].

References

1. Kosko, B., *Neural Networks for Signal Processing,* Prentice Hall, Englewood Cliffs, NJ, 1992, p. xiii.

2. Joseph, R.D., P.M. Kelly, and S.S. Viglione, "An Optical Decision Filter", *Proc. IEEE,* August 1963, pp. 1098-1118.

3. T.J. Sejnowski and P.S. Churchland, "Silicon Brains", *BYTE Magazine*, October, 1992, p. 137.

4. Mahowald, Misha and Carver Mead,"The Silicon Retina", *Scientific American*, May 1991, p. 76.

5. Jenkins, B.K., and A.R. Tanguay, "Photonic Implementations of neural networks", in Kosko, op. cit., pp. 287-382.

6. Farhat, N.H., et al., "Optical Implementation of the Hopfield Model", *Applied Optics*, 15 May 1985, pp. 1469-1475.

7. Jenkins, op. cit.

8. Dar-Shang, L., et al., "Bayesian and Neural Network Pattern Recognition: a Theoretical Connection and Empirical Results with Handwritten Characters", in *Artificial Neural Networks and Statistical Pattern Recognition*, I.K. Sethi and A.K Jain, eds., Elseviers Science Publishing, Amsterdam, 1991, pp. 89 - 108.

9. Khotanzad, A. and Jiin-Her Lu, "Shape and Texture Recognition by a Neural Network", Sethi, op. cit., pp. 109-131.

10. Hertz, J., et al., *Introduction to the Theory of Neural Computation*, Addison-Wesley, Redwood City, CA, 1991.

11. Chettri, S.R. and R.F. Cromp, "The Probabilistic Neural Network Architecture for High Speed Classification of Remotely Sensed Imagery", *1993 Goddard Conference on Space Applications of Artificial Intelligence*, pp. 119-132.

12. Kiang, R.K., H.H. Szu, and B.A. Telfer, "Classifying Multispectral Data by Neural Networks", *1993 Goddard Conference*, op. cit., pp. 169-177.

13. Fleming-Yure, Y.M., et al., "Data Fusion with Artificial Neural Network for Classification of Earth Surface from Microwave Satellite Measurements", *1993 Goddard Conference*, op. cit., pp. 145-154.

14. Ghosh, J., and A.C. Bovik, "Neural Networks for Textured Image Processing", Sethi, op. cit., pp. 133 - 154.

15. Stein, D.L., "Spin Glasses", *Scientific American*, July 1989, pp. 52-58.

16. Keller, J.J., "Crystals May Hold Key to New Computers", *Wall Street Journal,* January 5, 1993, p. B6.

17. Sengupta, D. and Iltis, R.A., "Neural Solution to the Multitarget Data Association Problem", *IEEE Trans. AES,* Vol. AES-25, No. 1, January 1989, pp. 96-108.

18. Szu, H.H., and R.L. Hartley, "Nonconvex Optimization by Fast Simulated Annealing", *Proc. IEEE,* Vol. 75, No. 11, November 1987, pp. 1538-1540.

19. Hertz, op. cit., p. 244.

20. Kumbla, K.K., J. Moya, and R. Baird, "Fuzzy Control of Robotic Manipulator", in *Fuzzy Logic and Control,* M. Jamshidie, N. Vadiee, and T.J. Ross, editors, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1993, pp. 292-328.

21. Hamilton, D.P., "Modest Benefits of Fuzzy Logic Become Clearer", *Wall Street Journal,* December 28, 1993, pp. B1-B2.

22. Nadler, M., and E.P. Smith, *Pattern Recognition Engineering*, John Wiley & Sons, Inc., New York, 1993, pp. 409-412.

23. Kasuba, T., "Simplified Fuzzy ARTMAP", *AI Expert*, November 1993, pp. 18-25.

24. Hung, Chuan-Chuan, "Building a Neuro-Fuzzy Learning Control System", *AI Expert*, November 1993, pp. 40-49.

25. Nadler, M., and E.P. Smith, *Pattern Recognition Engineering*, John Wiley & Sons, New York, 1993, p. 516.

26. Lippman, R.P., "An Introduction to Computing with Neural Nets", *Artificial Neural Networks: Theoretical Concepts*, IEEE Computer Society Press, Los Alamitos, CA, pp. 36-54.

27. Roth, M.W., "Neural Network Technology and its Applications", *Johns Hopkins APL Technical Digest,* Vol. 9, No. 3, 1988, pp. 242-253.

28. Hopfield, J.J., and D.W. Tank, "Computing with Neural Circuits: a Model" *Science* , 233, 1986, pp. 625-633.

29. Abu-Mostafa, Y.S. and J.M. St. Jacques, "Information Capacity of the Hopfield Model", *IEEE Trans. on Info. Theory,* 31(4), July 1985, pp. 461-64.

30. McEliece, R.J., et al., "The Capacity of the Hopfield Associative Memory", *IEEE Trans. on Information Theory* , 33(4), July 1987, pp. 461-82.

31. Lippman, op. cit.

32. Roth, M.W., "Survey of Neural Network Technology for Automatic Target Recognition", *IEEE Trans. NN*, Vol. 1, No. 1, March 1990, pp. 28-43.

33. Hertz, op. cit.

34. Kangas, J.A., et al., "Variants of Self-Organizing Maps", *IEEE Trans. NN*, March 1990, pp. 93-99.

CHAPTER 4. ARTIFICIAL INTELLIGENCE TECHNOLOGY TRENDS

SUMMARY

Fundamental theoretical questions about the nature of intelligence have yet to be answered, and are currently preventing the achievement of human-like intelligent behavior in machines. It is well-known that AI algorithms exhibit "brittleness", or the inability to perform outside narrow task domains. A minority view is that there are even more fundamental theoretical problems. It has been shown that human reasoning ability is not formalizable. And it has been persuasively argued that functionalism, the theory underlying symbolic AI, is logically untenable.

The development of a successful theory of AI requires progress along a broad front, including the fields of brain neurophysiology, the theory of computation, and even philosophy. The powerful critiques of the AI project that have emerged have been a useful stimulus for the re-examination of fundamental assumptions. The development of models of computation other than the Turing machine model, such as quantum computation, is one positive result. (A Turing machine is a mathematical generalization of the concept of a computing machine in terms of the simplest set of binary operations.) There has also been an expansion of research in alternatives to classical AI, such as neural networks.

However, the failure to find a theory of intelligence severely limits the application of AI in some fields. One obstacle to the practical use of AI applied beyond simple "blocks world" task domains is reliability, especially for control of high-value systems such as satellites, spacecraft, and ground stations.

1. INTRODUCTION

1.1 BACKGROUND

Integrated circuit technology is improving at a rate such that chip performance for a given scale size approximately doubles every 18 months. At this rate of improvement, one would expect that microcomputer chips in the year 2010 will have a computing performance of over 10,000 MIPS. The question arises as to what sort of applications such computers will be used for, aside from computationally-intensive scientific problems. For decades, there has been speculation about the possibility of "artificial intelligence", to the point where many now insist that it's inevitable. It is widely assumed, without proof, that there is smooth continuity between inanimate machine technology and the intelligence of living organisms, and that the achievement of intelligent performance in machines is a matter of incremental improvements in algorithms. However, some of the earlier aura of AI is wearing off, and people are now more reluctant to believe that progress in AI will be rapid.

This report examines the claims made by proponents of AI, and attempts to project the long-range future progress in AI based on that analysis. The focus is on symbolic inference systems, the most common variety of which are called expert systems. This represents the historical mainstream of AI research, and has direct application to NASA automated systems. Such techniques are already being applied by NASA and the Department of Defense for operational control of satellites. The goal is greater autonomy for space and ground vehicles and systems. This is especially important for robotic exploration of distant planets because of the long signal propagation times between the Earth and the other planets.

AI is a unique field from the engineering manager's point of view, because it suffers from an abuse of metaphor that tends to obscure the fundamental technical issues that can make or break a project. The abuse of terminology starts with the term "artificial intelligence". The president of Teknowledge, Jerrold Kaplan, once said that he wished the name had never been invented: "The words were designed twenty years ago to get money out of the government." A more appropriate term he suggested was "symbolic programming" [1]. The problem extends to some claims that are made as to what particular programs actually do. A prominent AI researcher, Douglas Lenat, made headlines in the late 1970s with a program called Automated Mathematician, and a later more sophisticated version called Eurisko. He claimed that these programs independently rediscovered the natural numbers, arithmetic, and key theorems of number theory. But these programs did not literally rediscover or prove theorems. The terminology is metaphorical. The programs functioned by generating symbol patterns that were syntactically correct according to a set of formal rules, and searched for "interesting" patterns by means of heuristics invented by the programmer. This is not to say that theorem-provers are not useful. The problem is rather how managers can assess actual achievements and future risk. A careful analysis of the actual functions of AI programs is required to accurately assess risk in AI research projects.

There are three broad approaches to the study of intelligence. At one end of the spectrum is the study of human and animal intelligence that includes neurobiology and psychology, and can be grouped as neuroscience. At the other end of the spectrum are those who believe that the human mind is in some fundamental way related to computers. This school itself covers a spectrum from those who believe that human beings are computers to those who believe that computers can be used for imitation of human capabilities. This is the research arena that has traditionally been called artificial intelligence. Bridging these fields is a discipline called cognitive science. Cognitive science puts primary emphasis on 'computational psychology' or how human cognition can be understood through computational analogies.

The three broad categories defined above are of interest because of their fundamental assumptions that influence the future course of technology. Of the three, AI is the engineering-oriented field, where the emphasis is on practical results. Yet paradoxically, AI is based on the most radical of hypotheses – that computers can do what humans do. It rests on a fundamental assumption, unique in science, that the mathematical models one develops for understanding the phenomenon under study – intelligence – are identical with the phenomenon.

The cognitive scientists focus on theories of human intelligence. They have formalized the fundamental AI assumption as the theory of 'functionalism'. Functionalists see mental processes as "rigorously specifiable procedures, and mental states as defined by their causal relations with sensory input, motor behavior, and other mental states. They view the brain as a computational system, asking what sorts of functional relations are embodied in it rather than which brain cells do the embodying, or how their physiology makes this possible" [2]. The theoretical basis of functionalism is further explained by Boden:

> "The theoretical groundwork... was provided by Alan Turing's 1936 paper on computable numbers which defined computation as the formal manipulation of (uninterpreted) symbols by the application of formal rules. The general notion of an 'effective procedure' – a strictly definable computational process – was illustrated by examples of mathematical calculation. But it implied that if intelligence in general is explicable in terms of effective procedures implemented in the brain, then it could be simulated by a universal Turing machine – or by some actual machine approximating thereto." [3]

Neuroscientists take the biological approach to intelligence. In practice this means that they regard the brain as a biological, chemical, and physical system that can be modeled as other real-world systems are modeled, but not necessarily that the system is a digital computational system, or that the model of the system is itself intelligent.

This paper focuses on a set of analyses by prominent thinkers as to why there has been a failure to find a general theory of intelligence. These analyses, it is hoped, shed light on what AI can and cannot do. This critical perspective can perhaps be of benefit to managers who will be responsible for funding research projects to enhance automation of NASA missions and operations.

1.2 DISCUSSION OF CURRENT AI RESEARCH

The mainstream of AI, herein referred to as symbolic AI, deals with symbolic representation of things and the algorithmic manipulation of the symbols to reach solutions to problems of inference. Systems utilizing this technique are commonly called expert systems, or knowledge-based systems. They are based on the premise that if one can capture the structure of the network of interconnecting causal and conceptual relations between words, concepts, and things, that structure can be programmed on a computer and used to calculate inferences from a set of input facts. Since the set of stored symbolic relations can be theoretically made very large, very complex models of the task-domain can be constructed.

This approach involves two components: 1) a database, or knowledge base, containing the names and attributes of the atomic entities constituting the world model or model of a sub-domain of the world, along with some judiciously chosen set of their relations to other atomic entities, and 2) an inference engine with which to search from a particular given set of facts to find useful new inferences.

1.2.1 The CYC Project

Some critics of symbolic AI claim that progress is lagging in symbolic inference systems, since all of those developed in the last two decades operate on the same basic principle. Rodney Brooks, of Massachusetts Institute of Technology's (MIT's) Artificial Intelligence Laboratory, for example claims that "most AI work is still done in the blocks world" [4], meaning that the task domains are still very narrow and restrictive. The blocks world was a simulated mini-world consisting of blocks of various shapes in an otherwise empty space. This conceptual world was used as the task-domain for several pioneering AI programs in the 1970s. The need to "break out of the blocks world" is the motivation for the CYC project.

The CYC Project [5], funded by the Microelectronics and Computer Technology Corporation (MCC), is Douglas Lenat's attempt to solve the problem of "brittleness" exhibited by present-day expert systems. Brittleness is the tendency of AI programs to function poorly when they are confronted with problems slightly outside the carefully controlled problem area for which they were designed. To do this he and his team are attempting to encode common-sense knowledge. The lack of common-sense knowledge has been held by most experts to be the reason for the brittleness of AI programs to date, although there is less than unanimous agreement that common-sense knowledge is actually knowledge that is symbolically representable. The central thesis of the CYC Project is that brittleness can be overcome by bringing to bear additional sources of knowledge. Its proponents argue that by broadening the knowledge base, the search for solutions is narrowed, because the constraints on the search are increased. They claim that ultimately they will incorporate most of what people know about the world, i.e. common sense.

The CYC Project is estimated to be a 1000 man-year effort, spanning two decades (1985-2005). In its first phase, a huge knowledge base is being constructed, along with about two dozen inference engine techniques, and hundreds of heuristic search rules, such as search by analogy. Lenat claims that about one million frames will be sufficient to allow the program to graduate to the second phase, in which it will begin "reading" and extracting knowledge on its own. These frames currently are being hand-coded into the program. They determine what knowledge is necessary by analyzing short pieces of text from common sources to determine the background knowledge required to understand the information in the text. In the second phase, frames supposedly will be generated automatically. A critique of CYC has been given by Brian Smith [6].

1.2.2 The Soar Architecture

One of the early pioneers in the field of AI was Allen Newell, who is now the prime mover behind what is called the Soar architecture for artificial intelligence [7]. This represents Newell's lifelong project to explain the nature of intelligence in terms of a single reductionist scheme.

Newell has been one of the chief supporters of symbolic inference processing, so his ideas are closely bound up with symbolic AI. His group's descriptions of Soar tend to approach it from a cognitive science point of view. That is, they see their symbolic processing technique as part of a hierarchy of functional levels that operate in human beings. Borrowing from psychology, they define the levels in terms of different time scales. The coarsest division is into three bands, in order of increasing time scales, the neural band, the cognitive band, and the rational band. These bands roughly correspond with research disciplines: symbolic AI falls within the rational band, while neural networks and neuroscience tend to deal with the neural level. The Soar architecture attempts to model the cognitive band, where processes operate on time scales from 10 msec to 10 sec. These processes include symbol accessing (memory), elementary deliberate operations, simple operator composition, and goal attainment. This schema represents a different "slice" of the space of possible models of systematic activity, one that closely models a plausible theory of low-level human mental activity, but that still relies on symbolic processing.

A detailed description of Soar is beyond the scope of this paper. However, a few essentials may be distilled. Soar is a system that seeks out a correct sequence of micro-actions required to reach a goal or solve a problem by means of trial and error search with learning. The learning involves not only declarative knowledge but procedural knowledge. Procedural knowledge is learned and grouped ("chunked") as sets of actions that have been successful in the past. This allows it to improve its performance when confronted with new example problems of the same class. It also incorporates other human-like capacities such as a kind of associative memory, an ability for context-switching, and an ability to back-track when it gets stuck. In summary, it is one of the most elaborate and carefully crafted symbolic inference systems devised to date. It is a very general architecture for problem-solving machines that attempts to mirror as closely as possible what is surmised about human mental processes. But it cannot yet be concluded that this architecture will outperform other approaches. It exhibits brittleness similar to that seen in other AI systems. The Soar group, like most practitioners of symbolic AI, makes some very basic assumptions about intelligence that have yet to be demonstrated, such as that mental processes may be reduced to formal procedures.

1.2.3 Rodney Brooks' Architecture

Rodney Brooks of MIT has advanced a general critique of symbolic AI, and has a research program based on what is termed a subsumption architecture to support his claims [8]. He says that symbolic processing, the attempt to encode representational models of task domains, is mistaken. Guided by his vision of how life and intelligence evolved in nature, he believes that human intelligence is an emergent property that is built on an enormous evolutionary sub-structure of non-representational knowledge. Based on this assessment he has attempted to construct robots on a non-symbolic architecture.

Non-representational knowledge is those things we know how to do without knowing how we do them. In Brooks' scheme of things, most human skills are non-formal, non-procedural; that is, they are not rooted in any set of explicit formal rules. Sensorimotor skills are the most basic form of this kind of knowledge, the mechanisms which evolution has spent vastly more time developing than it has higher level intelligence. For example, an insect may not know what an object is, but knows how to avoid running into it. Brooks proposes to build robots at the insect level of knowledge that know how to "walk", before building machines that can "think". He believes that in the process they can be made to develop skills that can be extended incrementally to higher levels of performance, ultimately up to the level of true intelligence.

In this architecture, the robot consists of multiple sensor and control subsystems, each with its own active process operating in parallel with all the others. The subsystems communicate with each other by means of very simple messages. There is no central representation or model of the world in this architecture. "It is a collection of competing behaviors. Out of the local chaos of their interactions there emerges, in the eye of the observer, a coherent pattern of behavior" [9].

Brooks' team has successfully built and tested robots that achieve a surprising degree of autonomy with simple software. They contain multiple microcomputers running asynchronously with respect to each other, each executing a low-level function, and communicating with other processors with short messages, usually a notification of a change of state among a few possible states. Brooks claims that these robots are the most autonomous robots developed to date, in the sense that they robustly operate in their environment continuously until their batteries run down. Other robots that have been developed for autonomous operation typically get stuck – the robot gets into situations the software cannot handle.

There are benefits to this architecture. First, it is potentially robust, because of its inherent parallelism: there is no central control function, and processing is distributed. This is an attractive feature for high-value robotic planetary exploration. Second, and perhaps most important, it does not depend on first solving the problem of knowledge representation, i.e. finding an accurate and complete model of the world. Brooks speaks of "indexing off the world"; that is, the robot takes its cues from the real world, not a model of it. Third, it may be easier to implement because the software is supposedly simpler. Brooks suggests a role for his architecture in planetary exploration. He sees small, lightweight robots conducting surface exploration in large numbers. Reliability would be doubly enhanced, first by virtue of the inherent fault tolerance, and second by virtue of numbers of duplicate machines. Such an architecture for planetary surface exploration would have an impact on communications.

Are Brooks' claims true? Researchers in neural networks confirm that learning complex adaptive behaviors without representation is possible. Neural network and connectionist architectures exhibit “recognition”, “learning”, “memory”, and complex input/output functions without internal declarative representations. Adrian Cussins [10] has analyzed this issue and theorizes how non-representational skills are possible. He hypothesizes how abstraction and conceptualization could arise from lower level skills, by means of what he calls perspective-independence. The question of representation is a key issue for AI, and one that Rodney Brooks has helped to bring to the fore. As with other schemes, subsumption architecture seems to capture some aspects of intelligent behavior, but there is currently no proof that it constitutes a general theory.

1.2.4 Connectionist Architectures

Connectionism, or the study of neural networks and related architectures is an alternative model of computation that can be used in the study of intelligence. Unlike symbolic inference, neural networks do not have local properties that are necessarily meaningful, it is the global properties that are important. A neural network is a mathematical function of discrete input and output variables, with a large number of internal parameters. The network is a general structure on which a particular function can be imposed by means of a "learning" process, by adjusting the internal parameters. That is, given a large "experience base", a set of inputs and their corresponding outputs, the internal parameters of the network, the connection weights, can be adjusted according to an automatic procedure, such that the network is then able to replicate the function implicit in the input/output pairs. (An analogy is a curve-fitting algorithm.) Neural networks have parallel structures and are well-suited to parallel computing, but can also be run on sequential machines using a serial simulation of the network.

Neural networks have been found to be useful for pattern recognition. They can pick out distinctions between patterns that have no simple analytical description. In this sense they are perhaps a better model for the feature-detection and discrimination capabilities of humans and animals. While a more complete description of neural network research is beyond the scope of this paper, we make note of the interesting fact that this set of problems being well-suited to a non-representational algorithm is evidence that intelligence is not solely a linguistic/symbolic process.

2. LONG-RANGE TECHNOLOGY TRENDS

In order to predict future progress, we need to assess the progress to date. Critics of symbolic AI claim that only little progress has been made since its beginning. A recent paper by William J. Clancey seems to confirm that all symbolic processing systems devised to date are variations of a single (decades-old) technique [11]. It appears that the trend in symbolic AI is a gradual discovery of its limits, rather than a discovery of major extensions and improvements. This trend is indicated by the massive shift in research interest toward neural networks in the 1980s.

The survey which follows reflects a significant and growing critical school of thought which seeks to illuminate the reasons for the failure of AI to live up to expectations.

2.1 ANALYSIS OF TRENDS

The often-stated goal of AI research is more "intelligent" computer programs, yet it is not entirely clear that symbolic AI is really about intelligence. If not, then it is perhaps unrealistic to posit ever-more intelligent machines in the future.

The assumptions of symbolic AI have been given a well-known formal definition by Alan Newell and Herbert Simon [12]. They hypothesized that "a physical symbol system is a necessary and sufficient means for intelligent action". This is based on the tenet of physics that no information can be transferred without a physical substrate, i. e., matter or energy (neglecting for the moment the information conveyed by the quantum wave function). On this foundation they build two additional theories, synopsized by Boden as a formal-syntactic theory of symbolism linked to a causal theory of semantics:

> “In their view, the criteria of identity of a symbol or computation are purely formal, and its significance is established by its causal history and effects. A symbol is a physical pattern, physically related to other patterns in various ways (such as juxtaposition) to form compound 'expressions'. Computational processes (realized by physical means) compare and modify patterns: one expression is the input, another the output. Any substrate physically capable of storing and systematically transforming expression-patterns can implement symbols. But for psychological purposes the substrate is irrelevant. To understand intelligence, we must describe physical-symbol systems at the information-processing level, in terms of designation and interpretation. These two semantic concepts are defined causally, such that the meaning of a symbol is the set of changes which it enables the system to effect, either to or in response to some internal or external state. The causal dependencies are essentially arbitrary, in the sense that any symbol could designate anything at all .” [13]

The key to the plausibility of this theory is the notion of causal semantics, that the meaning of an individual symbol arises from its relation to other symbols and to the context of its uses. The idea is empirically supported by the fact that the vast array of human culture is based on language, a system of arbitrary tokens standing for things and relations in the world. The linguistic analogy, sometimes called the Language of Thought theory, is another way of stating the physical-symbol system hypothesis. That is, there is an invisible symbol processing system underlying higher level mental functioning that mirrors language. Unfortunately, this formulation has been demonstrated to be logically untenable by prominent thinkers such as the mathematician-philosopher Hilary Putnam. This and other challenges to the assumptions of symbolic AI will be discussed in the subsequent sections.

2.1.1 John Searle and the Chinese Room

The philosopher John Searle has raised problems with the causal semantic theory of symbolic AI in several now famous articles [14,15,16]. His basic thought is that computers only manipulate symbols, they don't understand them. He offers as demonstration a thought experiment called the Chinese Room. Briefly, the argument runs:

a) Assume a formal procedure exists for giving intelligent answers to a human questioner (passing the Turing test); one that could be programmed on a Turing machine.

b) Assume the procedure is written as a series of steps in English.

c) Assume the questions and answers are conveyed in Chinese by means of Chinese symbol strings.

e) A human being could theoretically execute the procedure.

f) A human being who understands only English could execute the procedure.

g) Therefore it is possible to execute the procedure without understanding Chinese.

h) Therefore understanding of symbol meaning is not necessary to pass the Turing test.

The implication for AI is that understanding does not necessarily exist in a Turing machine simply by virtue of its having an algorithm that passes the Turing test, a result which would undermine the causal semantic theory. Searle argues that human beings are indeed instantiations of computer programs (the key tenet of functionalism), but that: a) they are instantiations of infinitely many computer programs, and b) so is everything else in the universe an instantiation of a computer program. This follows from the notion that any physical phenomenon can be represented as a set of numbers, and that there are infinitely many programs that could output a given set of numbers. But these are incidental facts. They do not necessarily explain the phenomenon. There is thus a category mistake in AI theory, according to Searle: the fact that a physical phenomenon has a mathematical representation does not mean that it is solely constituted by its mathematical properties.

2.1.2 Dreyfus and Models of the World

Stuart Dreyfus has analyzed AI from a somewhat different perspective. He argues that symbolic AI depends on a successful encoding of a formal model of the world [17]. This task, he maintains, has been attempted by philosophers over the centuries, for example Plato and Leibniz. In the twentieth century, the philosophers Ludwig Wittgenstein and Martin Heidegger spent decades attempting to come up with such a formalism, and independently came to the conclusion that it was impossible. Wittgenstein criticized his own work of decades earlier, the famous Tractatus Logico-Philosophicus : "Both Russell's 'individuals' and my 'objects' were such primary elements. But what are the simple parts of which reality is composed?...It makes no sense at all to speak absolutely of the 'simple parts of a chair'." Heidegger also found that the world could not be analyzed into a set of context-free elements. He observed that the common-sense background is a combination of skills, practices, and discriminations which do not have any representational content to be analyzed in terms of elements and rules. The conclusion from this is that there are no such things as context-free elements, or a complete world-model, only context-dependent partitionings that are used for some human purpose. This does not mean that partial world-models are not feasible for specific applications. But it does present a challenge to those attempting to construct exhaustive models like CYC.

The reaction of most AI researchers is that these are critiques by philosophers, and philosophers don't understand science or engineering. Daniel Dennett, one of the most prominent AI theorists, criticizes the philosophers for glossing over the hard problems. Something is going on behind the scenes in intelligent behavior; it is not an ineffable given. However, the case against the theoretical foundation of AI is also being made by former AI proponents, and Dennett himself has illuminated one of the field's most intractable philosophical/theoretical problems, as described in the next section.

2.1.3 Daniel Dennett and The Frame Problem

Dennett [18] describes the frame problem as figuring out what it is that humans do to focus on relevant facts and ignore irrelevant ones when engaged in common-sense reasoning. For example, if a robot were to successfully imitate human performance it would first have to be able to decide what actions will achieve its intended goal, then determine what unintended consequences might occur, and then determine which of these are undesirable. Unfortunately, since the list of candidates for possible unintended consequences is impossibly large, only the most relevant can be evaluated. So, additional knowledge for how to determine relevance must be added. But this additional knowledge can only be used if it is applied to all the candidates on the list, since their relevance is not known in advance. But the reason for the relevance test in the first place was to avoid having to evaluate the thousands of irrelevant effects.

The frame problem seems to be as fundamental as the brittleness problem. No one has a general solution to it, and no one knows whether there is a solution or not. The common practice is apparently to use ad hoc work-arounds and to patch up problems after they arise.

2.1.4 Drew McDermott and the Non-Formalizability of Inference

Related problems are discussed by McDermott [19]. He analyzes major shortcomings in logical methods for carrying out symbolic inference. It is well-known that inductive inference cannot be formalized. But McDermott points out other problems. It has become clear that general human reasoning is neither deduction nor induction. These are only very special cases of human reasoning ability. McDermott demonstrates why efforts to patch up logical methods using 'meta-theories' and 'non-monotonic logic' have also failed to rescue logical inference. There is no general theory of human reasoning ability, a fact which may turn out to be not merely due to no one being clever enough to discover it.

Are the solutions to these problems purely dependent on resolving centuries-old disputes in epistemology and logic? Or, are there mathematical equivalents of these problems that can be resolved by other means? At least one mathematician, Kurt Gödel, believed that many such philosophical questions could be resolved mathematically. His tool for doing so was the concept of encoding symbols as numbers, a concept that has since become widely known.

2.1.5 Kurt Gödel and Formal Systems

Gödel, Rucker, and Penrose, whose work was reviewed for this report, can be loosely grouped as having analyzed intelligence on mathematical grounds.

R. Rucker [20] describes the application of Kurt Gödel's famous Incompleteness Theorems to the problem of AI. Incompleteness refers to the theoretical incompleteness of any formal system: there are true statements arising within the formal system that cannot be proven to be so within the system. The notion of mathematical provability, and the deductive symbol processing methods based on it, are weaker than the notion of truth. One result is that it is impossible to design a formalism that encodes human mathematical intuition, thereby also implying that at least some other human creative faculties are not likely to be formalizable, and therefore not reducible to an algorithm.

Rucker has studied Gödel's work, as well as related work by other logicians and mathematicians, and finds related strict mathematical limitations on AI. For example, closely related to Gödel's theorems are the logical paradoxes known as Berry's Paradox, Richard's Paradox, and the Liar's Paradox. Rucker shows how Berry's Paradox mathematically demonstrates that there is no reasonably short algorithm that could encode the human faculty of turning words into concepts, that is, language understanding. By "reasonably short", he means an algorithm that is significantly shorter than a vast look-up table of special cases. More generally, a complexity-theory version of this argument by Gregory Chaitin [21] demonstrates that if a complex behavior can be described by a number (the Turing machine output modeling the behavior) then the algorithm causing that output in general must be as least as complex (be encoded by a number at least as large) as the output number. Rucker describes the implications of Richard's Paradox as showing that either the human ability for language understanding is infinitely complex, or is only describable by an intelligence more complex than human intelligence. Here Rucker defines language understanding as the conversion process from words to thoughts, which can be given precise mathematical definition as the formal procedure for translating a definition, a string of symbols, into a number.

The physicist Roger Penrose in his two recent books, The Emperor's New Mind and Shadows of the Mind, further elaborates on the Gödel-based critique, and presents new Gödel-type arguments [22,23]. He points to the ironic fact that Alan Turing, an early proponent of AI, used Gödel's work to show that there was a set of numbers not accessible by means of computation by a Turing machine, the set of which is the domain of non-recursive mathematics. He gives examples of problems in non-recursive mathematics about which it can be shown that there is no general formal solution, but for which humans can find particular solutions. The implication for AI is that there is no conceivable general algorithm for finding solutions to computational problems without assistance from human operators guided by mathematical insight. On the positive side, Penrose suggests the possibility that there are more general, non-Turing machine models based on quantum mechanics that may have more powerful properties applicable to problem-solving.

2.1.6 Alan Turing and Symbolic AI

Alan Turing invented the mathematical formalism of the universal Turing machine, which he used to demonstrate that there was no general algorithm for deciding all mathematical questions. He also showed that there was a class of numbers which he called computable numbers that can be calculated by a Turing machine, and another, more general set of numbers that cannot be so calculated. The former comprise the domain of recursive mathematics, while the latter are part of non-recursive mathematics. The key idea of AI was that any formal procedure whatsoever could be carried out by a Turing machine. The procedures which cause the machine to halt represent the computable numbers. Turing's hypothesis was that intelligence itself was a formal process of symbol manipulation. Since the universe of computable numbers is infinite, there are a lot of algorithms to be explored, leading to the fruitfulness of AI and computer science in general. However, the set of non-computable numbers has higher cardinality than the set of computable numbers. It therefore seems premature to suppose that intelligence is confined to the domain of recursive mathematics. All symbolic inference systems are symbol-string processing algorithms that have some Turing machine equivalent. Their possible outputs fall within the set of computable numbers. But how often will problems arise involving non-computable numbers? Are these rare, artificial problems? Roger Penrose points out that non-recursive problems are not rare or obscure. They crop up in simple string manipulations.

Perhaps the distinction between recursive and non-recursive mathematics mirrors the distinction between the blocks world and the real world. Penrose explores this question, seeking clues to the non-recursiveness of the real world in the laws of quantum physics, but provides no definitive answers, observing that research in this field is still young.

Robert Nadeau [24] also describes aspects of quantum mechanics that he argues should have a bearing on the science of intelligence, such as the non-locality of the quantum wave function. Like Searle and Penrose, Nadeau considers the distinction between mathematics as a representational language for describing reality, and reality itself. He points out a hidden assumption at the foundation of AI – that mathematical descriptions can completely and exhaustively capture the nature of physical phenomena. Since quantum physics shows, according to Nadeau, that this is not a valid assumption, mathematical models cannot necessarily duplicate the physical causes of intelligence.

2.1.7 Hilary Putnam and Functionalism

Hilary Putnam of Harvard has studied these issues for over forty years, and was one of the originators of functionalism. He is in a unique position to evaluate the problem, being expert in mathematical logic, physics, and philosophy. Putnam has criticized Roger Penrose's arguments, not because he disagrees with Penrose's assessment of AI, but because of what he perceives to be technical flaws in the arguments. Although Putnam was once one of the foremost proponents of functionalism, he now repudiates that position. Putnam brings to bear an array of arguments against functionalism and the possibility of AI. In The Project of Artificial Intelligence, Putnam summarizes his views. [25]

Putnam states that the original impetus to the hypothesis that intelligence is computation was the idea that behavior can be modeled numerically as physical actions, and therefore as the output of some Turing machine. This idea is flawed, he claims. One problem, which was not known at the time when he worked on the theory of functionalism, is that physical systems do not necessarily behave according to Turing-computable (recursive) functions. (This fact also gave impetus to Roger Penrose's writings.) Secondly, the numerical aspect of behavior is incidental. The numerical model does not provide a perspicuous representation. This is in keeping with his overall skepticism about the possibility of reducing human behavior and its meanings to something in which meanings and purposes have no part, such as physics or computation.

If there is no theoretical reason to think that the mind is a Turing machine, is there nonetheless a reason to think that a computer could simulate intelligence? Putnam discusses the problems with formalizing induction mentioned in section 2.1.4. What is the scope of induction? When do you know when you have enough samples? How do you know when to rely on previous experience, and when not to? How do you resolve conflicting chains of induction? These are some of the practical questions regarding the nature of induction. Another problem is the seemingly unlimited number of ways that humans classify things as similar that have no basis in a reductive domain such as similarity of sensory stimuli. Many examples of similarity-judgment depend on a common background of human purposes. Again we run into the problem of common sense, or how much of intelligence presupposes the rest of human nature. Natural language understanding is another fundamental aspect of intelligence that does not appear to be one single function. Like induction, it is a vast complex of abilities that would require generations of researchers to formalize, if at all. Putnam believes there is not likely to ever be a definitive answer to the question as to whether AI is feasible, but that in any case the AI project is "utopian".

2.1.8 David Marr and Complexity

Closely related to the question of the duplication of physical phenomena in computer models is the issue of system complexity. David Marr [26] distinguished two types of natural systems – those that have a concise or well defined mathematical theory, and those that are too complex to have such a theory. He held that symbolic processing systems were incapable of modeling higher level mental processes because human intelligence is of the second kind. There is thus the possibility that no general theory of intelligence is possible.

2.2 FUTURE APPLICATIONS

The critical analysis of the foundations of symbolic AI will lay the groundwork for future progress. A fundamental requirement for the use of AI in high-value applications is improved methods of estimating reliability of the software. A better understanding of the theoretical foundations of AI will permit progress in understanding reliability.

Symbolic inference processing has many practical applications. There has been steady progress in finding ingenious ways to formalize and model restricted task-domains. In the 1980s expert systems moved from university laboratories to commercial applications. The classic application is medical diagnosis, such as provided by the programs MYCIN and INTERNIST. These programs operate from a base of a few hundred heuristic rules encoding expert medical knowledge.

The extension to diagnosis of electronic systems is clear. In the realm of aerospace engineering, expert systems have been applied to fault diagnosis for satellites. A recent example was a satellite communications network diagnostic prototype developed by SAIC that used a neural network to classify data error patterns on the front end, interfacing to an expert system to infer fault hypotheses from the error patterns [27]. Similar applications are described in papers presented at the International Symposium on Space Information Systems [28].

Some of the topics of research in AI that have yielded practical results are expert systems, machine recognition, and neural networks. All of these topics may produce applications to NASA missions. More specifically, AI has direct application to communications. Some example applications of expert systems and neural computation to NASA's OSC are: autonomous satellite and ground systems, fault tolerance, real-time diagnostics, imagery compression, adaptive signal processing, dynamic traffic routing and link control, and optimal scheduling. Expert systems and neural networks can be applied to various optimization problems that crop up in communications such as network traffic routing, scheduling, and adaptive data compression and signal processing. The marriage of these two AI approaches can be fruitful, since they often can be applied to two different parts of the same problem. The neural network is useful for reducing large amounts of data to discrete classes to be operated on by symbolic processing.

2.3 EMERGING TECHNOLOGIES

There do not appear to be any imminent and dramatic advances or clear trends for technology in the domain of AI algorithms, only incremental improvements. The field of AI is currently in a stage of developing tools for automation of complex tasks, inspired by, but not necessarily based on human intelligence. The future of AI should therefore be seen not as the eventual achievement of intelligence in the human sense, but rather the improvement of classifiers, recognizers, optimizers, knowledge-bases, and other tools for automation.

In the last decade, neural networks have emerged as a powerful tool for pattern classification based on training. Neural network architectures are well suited to fine-grained massively parallel computing. Thus, the advance of solid-state electronics will enable the implementation of more powerful real-time neural networks. However, this is not a new conceptual advance.

The adaptive capabilities of neural networks and the evolutionary aspects of genetic algorithms reflect an increased interest in an evolutionary model. This will be an important new approach to autonomous systems research in the future. Nanotechnology, the science of constructing machines at the nanometer scale, will make use of biological models of self-replication and other capabilities of microorganisms. The idea is to replicate capabilities of primitive life forms before attempting to replicate higher ones. Micro-miniature autonomous machines could be used for the exploration of Mars. The use of "microrobots" based on ideas like the subsumption architecture has been extrapolated into the long-range future to the case of self-replicating machines. The concept of a self-replicating machine was explored by the mathematician John von Neumann. The requirements are complex, and are beyond the present-day state of the art. The most likely direction for meeting the requirements is construction of machines at the molecular level. Nanotechnology is a concept that is beginning to be explored, and may bear fruit several decades hence. Exploration of a planetary surface by means of self-replicating molecular machines would be carried out by means of a process of exponential expansion in numbers, using materials found on the planet's surface, similar to biological growth. Alternatively, microorganisms could be genetically modified to live in hostile planetary environments, and to perform information gathering functions.

3. RESULTS

3.1 ,IMPACT OF ARTIFICIAL INTELLIGENCE TECHNOLOGY ON NASA

A picture emerges in which brittleness may turn out to be the result of inherent theoretical limitations of symbol processing systems, limitations that cannot be entirely eliminated. This means that the reliability of expert systems may not be adequate for completely unattended operation of AI-based autonomous systems in complex environments. More research needs to be done on symbolic processing systems in order to more rigorously characterize what are their limitations. This survey also shows that critical analysis is important. In this field, perhaps uniquely, critical philosophical reasoning plays a major role in showing how the assumptions of a scientific/engineering enterprise are either valid or invalid. Seemingly innocent assumptions made by computer scientists as to the nature of intelligence have been shown to be false by counter examples discovered by philosophers such as Putnam.

3.1.1 Automation

The use of AI within NASA did not get underway until the mid-1980s, when the Artificial Intelligence Research and Development Program was initiated to support the Space Station [29]. The first successful application of AI was an expert system called INCO (for integrated communications officer), at Johnson Space Flight Center. It was an AI-assisted mission control console that made extensive use of modern graphics and rule-based fault-diagnosis. INCO achieved a great speed-up in the detection of problems, and eventually led to a reduction of staff at the control center. A similar system called the spacecraft health automated reasoning prototype was implemented at JPL for the telecommunications subsystem of the Voyager spacecraft, and was later applied to the Magellan and Galileo missions. A system called knowledge-based autonomous test engineer – liquid oxygen was being tested as of 1992 at Kennedy Space Center. It is for monitoring the loading of liquid oxygen into the fuel tank of the shuttle. It was able to identify problems well in advance of human operators working with the same telemetry.

Since these pioneering efforts, there has been a great proliferation of AI-based fault monitoring and diagnosis systems, both in academic research and in real space systems. An example is the Selective Monitoring System. This system uses model-based reasoning to determine which parts of a system are causally upstream and downstream from a failure to help technicians to focus more quickly on the problem [29].

This kind of system is able to speed up and simplify the work performed by human operators, and thereby improve productivity. However, no expert system used in a critical application has been able to reach a level of sophistication that would enable a system to be operated in a completely unattended manner. Rather, they are used to support and enhance human activities.

AI systems can provide confirmation or expert second opinion in mission critical situations. The problem with this strategy is that human beings in this situation may tend to avoid contradicting the machine and place too much reliance on it. The machine can always be blamed if a mistake is made. As analyzed by Joseph Weizenbaum [30] reliance on expert systems has the consequences "first that decisions are made on the basis of rules and criteria that no one knows explicitly, and second that the system of rules and criteria become immune to change." In order to avoid inappropriate reliance on a machine, the user would have to question every decision and carry out his own analysis of why the machine made the decision it did, or study a printout of the machine's inference chain. Such a situation would negate benefits in the speed-up of decision making.

In order for expert systems to be used in mission-critical applications in an unattended mode, a great deal more reliability would be required than is currently available in AI systems, since AI-based control is, in a sense, non-deterministic. The high value of the spacecraft would necessitate that AI-based control be limited to those aspects of spacecraft operations that cannot be coped with through remote control, since it is unlikely that all decision making would be turned over completely to the on-board software. For planetary exploration and other far-term space missions, spacecraft autonomy will be necessary because of the radio propagation delay times. The historical track record of deep-space missions shows that a high degree of autonomy can be had with non-AI control techniques, owing largely to the relatively simple and stable environment of deep space, and the predictable control regime. Use of expert systems could extend and enhance the autonomy of spacecraft, particularly in the area of on-board fault diagnosis and repair. Unmanned orbital maneuvering and docking procedures in planetary orbit would also require at least some level of AI automation.

Unmanned planetary surface exploration by robotic rovers would be impractical without a high degree of AI-based autonomy. The kind of autonomy required by rovers is different than for spacecraft because the environment is more complex. Getting from point A to point B requires navigating through a maze of obstacles, and presents the robot with a continuous stream of contingencies. Several kinds of AI are applicable here, including expert systems. Expert systems can be used for scene image understanding, data understanding, route planning, and recovery from contingencies. However, improved robustness will be needed to mitigate the brittleness problem. Alternative, non-symbolic architectures such as the subsumption architecture may help in this. Rodney Brooks proposes an architecture consisting of a large number of small, low-cost robots, each carrying out simple functions and objectives. The failure of some percentage of the robots would not curtail the entire mission. His subsumption architecture for robot design is also supposed to reduce the complexity and number of single point failures for the robots themselves.

Expert systems can also be used for planning routes and navigation, for getting out of difficult situations (getting unstuck), and for carrying out data analysis on site. Neural networks are also applicable to on-board data analysis that is not amenable to explicit formal rules. Neural networks can be used for feature extraction and pattern recognition, particularly when the features or patterns are implicit in the data, and must be extracted from a large training set. On-site data analysis can result in large communications data rate reductions.

3.1.2 Communications

Analysis of images to recognize obstacles, the most basic requirement for autonomous robots, can result in a spin-off that directly impacts communications functions. Aside from the reduced control requirement provided by vehicle autonomy, the decomposition of images into objects can be used for image compression. An early example of this is the type of compression that uses edge detection to transmit only object edges. At a higher level of abstraction, in which objects such as rocks are situated in a 3-dimensional space, the objects can be modeled as simpler geometric shapes, and only the equations for these shapes are transmitted.

AI could be applied to an adaptive bandwidth communications scheme. The strategy would be to optimize bandwidth utilization by switching between different modes. For example, for imagery, one could switch between high resolution and low resolution, between high compression ratios and low compression ratios, between high and low frame rates, etc. AI would be used to decide when to switch. For example, AI could be used to decide when the scene appearing in the field of view is "interesting" in some scientific sense, so that a high resolution frame could be transmitted. Adaptive bandwidth selection will become increasingly important as the gap between lossless compression and lossy compression techniques widens. It is currently estimated that lossy compression techniques for imagery can be as high as 1000-to-1. Channels containing signals of drastically varying bandwidth affect the design of a communications architecture.

AI can be used in signal processing. For example, the classic problem is enhancing the probability of detection while reducing the false alarm rate with a given signal to noise ratio.

Signature recognition techniques can accomplish this, and AI is a natural fit to this problem. There is a school of thought that neural networks coupled to expert systems can solve real-world signature problems more effectively than other approaches. One communications application of signature recognition is the characterization of link problems from bit error rate characteristics.

AI can be used for communications network management. It can serve functions of fault monitoring, fault detection, fault isolation and fault recovery. This extrapolates to knowledge-based control of global network attributes such as optimization of allocation of network resources. AI-based diagnostics are the key to minimally attended ground processing facilities. Automation of installation, configuration and management of communications networks is a mundane but high payoff application of AI. Some current products apply AI at the level of LAN analyzer devices, in effect portable expert systems. Even at this level, AI can reduce training time, skill level, and repair time.

One of the characteristics of intelligence is that it is adaptable to different contexts. This is the principle behind the concept of smart network navigation "agents" that autonomously seek out the optimum network path, sense channel and network path characteristics, and adjust data compression, error codes, packet formats, and packet encapsulations accordingly. The agent could propagate software "sensors" to gather the required information from various nodes in the network. This technology would essentially eliminate the problem of interface compatibility and standards. The software agent would essentially be an expert system on communications protocols and codes. A protocol expert system would be equivalent to a universal communications protocol.

3.1.3 Cost Savings and Productivity

The question of the potential cost-benefits of applying AI to OSC operations is part of a larger question of the cost-benefits of automation in general. A recent study for the General Services Administration called Beyond FTS 2000 discussed this at length [31]. A key observation of this report was that the number of federal employees has not changed in twenty years. Increases in workloads beyond what existing staff levels can handle have typically been performed by contractors. While many agencies will be forced to operate with flat or minimally rising budgets in coming years, the expected workload and responsibilities of these agencies will be increasing at a much faster rate. This is exactly the case with NASA as it carries out its mandate to provide data on the global environment. The implication is that cost reduction is not the issue so much as productivity improvement. That is, a fixed level of staff will be responsible for more and larger tasks. Logically, it would seem that any automated system that takes over routine functions must free up man-hours for other purposes, resulting in increased productivity. This is very difficult to demonstrate empirically, however. It is argued by most economists that 1 trillion dollars spent on computer automation in the last decade has not resulted in increased worker productivity. Productivity in the services sector actually declined. Some have explained this as due to the fact that work expands to fill the capacity available. In other words, as workers' capabilities using computers increased, their workloads increased. According to this view, the new workload must therefore involve a large amount of non-productive activities. Without more definitive long-term cost-benefits data it is difficult to justify the high current costs of AI for productivity improvement.

Artificial intelligence can obviously play a role as part of an overall strategy of automation. Unfortunately AI is very expensive, and is still unproven outside a few specialized areas. Because of fundamental theoretical uncertainties about the capabilities of AI, especially in regard to reliability, it is very difficult to extrapolate to the future, or to recommend it as a solution to problems of automation. It is anticipated that automation will improve by incremental steps using a mixture of standard computer automation and more advanced AI techniques. Looked at as a set of heuristic problem solving tools, AI applications will expand and broaden. However, it is unrealistic to expect AI to exhibit intelligent performance in the sense of human intelligence within the next 20 years. Rather, AI will provide productivity gains in the operations and maintenance domain.

3.2 TECHNOLOGY ROADMAP

A technology roadmap, or possible timeline, of major developments in the field of artificial intelligence over the next 25 years follows.

1996-2000: In the past twenty years, symbolic AI techniques have been improved, refined, and extended, but the basic techniques remain the same. In the next five years, progress will likely be limited to incremental refinements of these techniques. Projects such as CYC may begin to show results, that is, mimicry of common sense to some degree. These will operate on large-scale models of the everyday world. The first such systems are now being developed at great expense. There will continue to be a proliferation of research directions in AI beyond the confines of symbolic AI. Neural networks, fuzzy logic, and genetic algorithms seem to be alternative architectures that currently are in high favor. A common theme is methods that deal with processes in the real world that do not have explicit or linguistic representations, and that involve learning. Some observers think that neural networks, fuzzy logic, and genetic algorithms are the best approach to solving problems involving non-analytic classification, uncertainty, incompleteness, and ambiguity. These techniques do not exhibit brittleness (at least in the same way), but rather smooth transitions and partial solutions. In many cases, especially in real-time planning and optimization systems, partial solutions and best guesses are often adequate. Human beings and other animals get by in this way in most situations. One author, Nadeau, believes this an area that will see a quantum jump in applications in the near future.

2000-2010: General-purpose world-models may in the long term be embedded in a wide variety of electronic systems, taking advantage of the increasing processing power, and lower cost of integrated circuits. As processing power increases, heuristic search methods used in knowledge-based systems become more cost-effective for more applications. One can foresee a time when the CYC knowledge base or the entire Library of Congress will reside in a hand-held expert system. The expert system will be a "knowledge agent" retrieving information from the database. Trends for storage and processing technology indicate that this will be feasible before 2010 [32].

Systems that mimic learning and evolution will be increasingly important. The subsumption architecture is probably the first phase of a long period of experimentation in "artificial life" or "artificial evolution". If David Marr is correct in thinking that human intelligence is too complex to have a theory, then an evolutionary, or "bottom-up" approach is probably superior to the symbolic, or "top-down" approach.

2010-2020: It could be argued that the missing element in all forms of AI today is sentience, the ability to feel the environment. Sentience is a property of living things that seems to have a physical substrate, but at present cannot be deduced from that physical substrate. Common sense would tell us that sentience is the motivator of living organisms, and that the ability to feel different qualities in the environment allows them to navigate in that environment. If sentience cannot be designed into machines, then perhaps living matter can be spliced into machines. Hybrids of living matter and machines are a possible direction for future research, although such research may raise ethical issues.

3.3 FEASIBILITY AND RISK

This report has described factors important to the success of future AI projects and applications. The results suggest that it may be possible to develop a risk-assessment methodology based on several key factors. At the top of the list is identification of a theory underlying the proposed program. Does such a theory exist, or is the approach purely heuristic? Does the program have potential problems like the frame problem? If so, what are the proposed remedies? The results of a theoretical evaluation will determine the next important risk-assessment factor: reliability. AI algorithms with no underlying mathematical theory will be ranked at a lower reliability level than ones that have such a theory. The application for which the proposed AI system is appropriate will be determined by what class of reliability it has. And finally, there would be a cost-benefits analysis. This would take into account the fact that incremental advances in AI may require geometric increases in funding.

Leading-edge AI will in the future be concerned with broad-based knowledge systems, very large databases, and generalized world-models. The cost of developing these systems will be high, perhaps too high for a single federal agency operating on a near-flat budget. It is anticipated that a cut-back in defense spending will impact the progress of AI, since DoD has been a traditional supporter of AI. There has also been a somewhat lessening enthusiasm for AI as the initial glamour has worn off, and as many promises have failed to materialize, such as natural language understanding. The generalized nature of new AI systems like CYC is an incentive for cooperative ventures in the form of industry consortia or multi-agency financing within government as traditionally done by DoD's Advanced Research Projects Agency.

It is expected that reliability will be the chief obstacle to the deployment of AI within NASA, since reliability is critical for NASA's mission success. AI software suffers from the brittleness problem. One solution now being proposed, the CYC project, is an expensive solution, and not one that will be in widespread use for many years. It is also not clear that CYC will not have its own set of unique problems that will take years more to debug.

References

1. G. Johnson, *Machinery of the Mind*, New York: Random House, 1986, p. 237.
2. M. Boden, ed., *The Philosophy of Artificial Intelligence*, New York: Oxford University Press, 1990, p. 2.
3. *Ibid.*, p. 4.
4. R. A. Brooks, "Intelligence Without Representation", *Artificial Intelligence*, Vol. 47, Jan. 1991, pp. 139-159.
5. D.B. Lenat and E.A. Feigenbaum, "On the Thresholds of Knowledge", *Artificial Intelligence*, Vol. 47, Jan. 1991, pp. 185-250.
6. B. Smith, "The Owl and the Electric Encyclopedia", *Artificial Intelligence*, Vol. 47, Jan. 1991, pp. 251-288.
7. P.S. Rosenbloom, et. al., "A Preliminary Analysis of the Soar Architecture as a Basis for General Intelligence", *Artificial Intelligence*, Vol. 47, Jan. 1991, pp. 289-325.
8. Brooks, *op. cit.*
9. Brooks, *op. cit.*, p. 149.
10. A. Cussins, "The Connectionist Construction of Concepts", in Boden, *op. cit.*, pp. 368-440.
11. W. J. Clancey, "Model Construction Operators", Artificial Intelligence, Vol. 53, Jan. 1992, pp. 1-115.
12. A. Newell and H. Simon, "Cognitive Science as Empirical Enquiry: Symbols and Search", *Communications of the Association for Computing Machinery*, Vol. 19, March, 1976. (Reprinted in Boden, *op. cit.*)
13. Boden, *op. cit.*, p. 8.

14. J. R. Searle, "Minds, Brains, and Programs", in Boden, *op. cit.*, pp. 67-88.

15. J. R. Searle, *Minds, Brains and Science*, Cambridge: Harvard University Press, 1984.

16. J. R. Searle, "Is the Brain's Mind a Computer?", *Scientific American*, January 1990, p. 26.

17. H.L. Dreyfus and S.E. Dreyfus, "Making a Mind Versus Modelling the Brain: Artificial Intelligence Back at a Branch-Point", in Boden, *op. cit.*, pp. 309-333.

18. D.C. Dennett, "Cognitive Wheels: The Frame Problem of AI", in Boden, *op. cit.*, pp. 147-170.

19. D. McDermott, "A Critique of Pure Reason", in Boden, *op. cit.*, pp. 206-230.

20. R. Rucker, *Infinity and the Mind*, Boston: Birkhauser, 1982.

21. G. Chaitin, "Randomness and Mathematical Proof", *Scientific American*, May 1975, pp. 47-52.

22. R. Penrose, *The Emperor's New Mind*, New York: Oxford University Press, 1989.

23. R. Penrose, *Shadows of the Mind*, New York: Oxford University Press, 1994.

24. R.L. Nadeau, *Minds, Machines, and Human Consciousness*, Chicago: Contemporary Books, 1991.

25. H. Putnam, *Renewing Philosophy*, Cambridge, MA: Harvard University Press, 1992.

26. D.C. Marr, "Artificial Intelligence: A Personal View", in Boden, *op. cit.*, pp. 133-146.

27. SAIC Report on SECONNS (a neural network-based satellite communications diagnostic system), FEDSIM Contract no. MDA903-87-D-9012.

28. Penrose, *op. cit.*, p. 130.

29. Montemerlo, M.D., "The AI Program at the National Aeronautics & Space Administration", AI Magazine, Winter 1992, pp. 49-61.

30. Nadeau, *op. cit.*, p. 97.

31. B.J. Bennington, "Beyond FTS2000: A Program for Change", GSA Contract No. GS00K88AFC1200, for the National Academy of Sciences, 1989.

32. SAIC Report, "Alternative Technology Assessment", FEDSIM Contract No. MDA903-87-D-9012, for NASA Office of Space Operations, Nov. 1991.

ANNEX 1. ACRONYMS AND ABBREVIATIONS

AI	Artificial intelligence
ANDF	Architecture-neutral distribution format
ARPA	Advanced Research Projects Agency
ART	Adaptive resonance theory
ATM	Asynchronous transfer mode
CAM	Content-addressable memory
CASE	Computer-aided software engineering
Corba	Common object request broker architecture
CPU	Central processing unit
DBMS	Database management system
DOC	Digital optical computer
DoD	United States Department of Defense
EOS	Earth observing system
FCS	Fiber channel standard
4GL	Fourth-generation (programming) language
FTS 2000	Federal telecommunications system 2000
Gbps	Gigabits per second
GUI	Graphical user interface
GQM	Goal-question-metric
HIPPI	High performance parallel interface
INCO	Integrated communications officer
I/O	Input/output
IS	Information system
JPL	Jet Propulsion Laboratory
LAN	Local area network
LED	Light-emitting diode
Mbps	Megabits per second
MIPS	Million instructions per second
MIT	Massachusetts Institute of Technology
NASA	National Aeronautics and Space Administration
NCSA	National Center for Supercomputer Applications
OFDM	Optical frequency division multiplexing
OLTP	On-line transaction processing
OMG	Object Management Group
OOP	Object-oriented programming
OSC	Office of Space Communications
OSF	Open Software Foundation
PC	Personal computer
PTD	Parallel transfer disks
RAID	Redundant array of inexpensive disks
RAM	Random access memory
RISC	Reduced instruction set computer
SAIC	Science Applications International Corporation
SFAM	Simplified fuzzy ARTMAP
SIMD	Single-instruction, multiple-data
SLM	Spatial light modulator
SONET	Synchronous optical network
3GL	Third-generation (programming) language
TQM	Total quality management
UNIX	De-facto standard multi-tasking operating system for workstations
VLSI	Very large scale integration

WAN	Wide area network
WDM	Wavelength division multiplexing

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE May 1996	3. REPORT TYPE AND DATES COVERED Final Contractor Report

4. TITLE AND SUBTITLE Technology Directions for the 21st Century Volume II	5. FUNDING NUMBERS WU–315–90–81 DAEA32–96–D–0001
6. AUTHOR(S) Giles F. Crimi, Henry Verheggen, John Malinowski, Robert Malinowski, and Robert Botta	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Science Applications International Corporation 1710 Goodridge Drive McLean, Virginia 22102	8. PERFORMING ORGANIZATION REPORT NUMBER E–10241
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) National Aeronautics and Space Administration Lewis Research Center Cleveland, Ohio 44135–3191	10. SPONSORING/MONITORING AGENCY REPORT NUMBER NASA CR–198479

11. SUPPLEMENTARY NOTES

Project Manager, Denise S. Ponchak, Space Electronics Division, NASA Lewis Research Center, organization code 5610, (216) 433–3465.

12a. DISTRIBUTION/AVAILABILITY STATEMENT Unclassified - Unlimited Subject Categories 32, 17, 33, 60, and 76 This publication is available from the NASA Center for AeroSpace Information, (301) 621–0390.	12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

Database applications are sufficiently general and address a large enough market so that it is both feasible and economical to build specialized hardware and architectures to accelerate database processing. Because large databases may have a large number of users, the requirement for processing power and high-speed storage access may be greater than that allowed by current technology. Several technologies are available or on the horizon that could be used to accelerate database processing. These include: parallel arrays of magnetic disks for access time improvement; holographic imaging for parallel access of optical disks and photorefractive crystals; associative memory for parallel access; balanced systems architectures, or architectures in which processing, storage and networking are well-matched for maximum throughput; photonic devices and architectures for database acceleration; and artificial intelligence (AI) technology applied to database search.

14. SUBJECT TERMS Data base; Software; Technology; Neural networks; Artificial intelligence	15. NUMBER OF PAGES 77
	16. PRICE CODE A05

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT